



Delft University of Technology

Highway Traffic Congestion Patterns Feature extraction and Pattern retrieval

Nguyen, T.T.

DOI

[10.4233/uuid:735fe56b-c6ae-4432-ad80-6da39d3666ed](https://doi.org/10.4233/uuid:735fe56b-c6ae-4432-ad80-6da39d3666ed)

Publication date

2021

Document Version

Final published version

Citation (APA)

Nguyen, T. T. (2021). *Highway Traffic Congestion Patterns: Feature extraction and Pattern retrieval*. [Dissertation (TU Delft), Delft University of Technology]. <https://doi.org/10.4233/uuid:735fe56b-c6ae-4432-ad80-6da39d3666ed>

Important note

To cite this publication, please use the final published version (if applicable).
Please check the document version above.

Copyright

Other than for strictly personal use, it is not permitted to download, forward or distribute the text or part of it, without the consent of the author(s) and/or copyright holder(s), unless the work is under an open content license such as Creative Commons.

Takedown policy

Please contact us and provide details if you believe this document breaches copyrights.
We will remove access to the work immediately and investigate your claim.

Highway Traffic Congestion Patterns: Feature Extraction and Pattern Retrieval

Thiện Tín NGUYỄN

This doctoral dissertation was funded by the Dutch National Data Warehouse of Traffic Information (NDW). It was also part of the MiRRORS project (with project number 16720) within the Open Technology Program, which is (partly) financed by the Applied Sciences Division of the Dutch Research Council (NWO/TTW).

ndw



Cover photo: Joey Kyber

Highway Traffic Congestion Patterns: Feature Extraction and Pattern Retrieval

Dissertation

for the purpose of obtaining the degree of doctor
at Delft University of Technology
by the authority of the Rector Magnificus, Prof.dr.ir. T.H.J.J. van der Hagen,
Chair of the Board for Doctorates
to be defended publicly on
Friday 16 July 2021 at 12:30 o'clock
by

Thiện Tín NGUYỄN

Master of Engineering in Computer Science,
Ho-Chi-Minh City University of Technology, Vietnam
born in Quang Tri, Vietnam

This dissertation has been approved by the promotor:
Prof.dr.ir. J.W.C van Lint, Prof.dr. H.L. Vu

Copromotor: Dr.ir. S.C. Calvert

Composition of the doctoral committee:

Rector Magnificus,	chairperson
Prof.dr.ir. J.W.C van Lint,	Delft University of Technology, promotor
Prof.dr. H.L. Vu,	Monash University, promotor
Dr.ir. S.C. Calvert,	Delft University of Technology, copromotor

Independent members:

Dr. E.I. Vlahogianni,	National Technical University of Athens
Prof.dr.ir. C.M.J. Tampère,	Katholieke Universiteit Leuven
Prof.dr. E. Eisemann,	Delft University of Technology
Prof.dr. S. Ahn,	University of Wisconsin - Madison
Prof.dr.ir. S.P. Hoogendoorn,	Delft University of Technology, reserve member

TRAIL Thesis Series no. T2021/22, the Netherlands Research School TRAIL

TRAIL
P.O. Box 5017
2600 GA Delft
The Netherlands
E-mail: info@rsTRAIL.nl

ISBN: 978-90-5584-297-1

Copyright © 2021 by Thiện Tín Nguyễn

All rights reserved. No part of the material protected by this copyright notice may be reproduced or utilized in any form or by any means, electronic or mechanical, including photocopying, recording or by any information storage and retrieval system, without written permission from the author.

Printed in the Netherlands

*Patience is the mother of success.
Kiên nhẫn là mẹ thành công.
(Vietnamese proverb)*

Acknowledgements

Study abroad was a big dream of mine. Finally (also fortunately), that dream came true; June 16, 2016, marks the beginning of my PhD journey in the beautiful city, Delft, the Netherlands. It has been a long adventure filled with many great things that I would like to acknowledge.

I cannot express enough my tremendous gratitude to my promotor, Prof. Hans van Lint, who has constantly supported me throughout the five years. I gratefully appreciate the most his guidance and (unbelievable) confidence in me. There is a proverb saying "good teachers teach, great teachers inspire". That simply shows how fortunate I was to have his supervision. Every single discussion with him is a privilege. His enthusiasm and inspiration are amazing, and certainly a fortune for (PhD) students. The frequent meetings with my daily supervisor, Dr Simeon Calvert, keep my work 'more or less' better structured (I did try, Simeon!). Not only Simeon guides me with scientific research, but he also was there to mentally support me through an extremely hard time during the first half of my PhD. I am and will always appreciate that very much. While Hans and Simeon can give me inputs from the traffic domain, I am glad to have Prof. Hai L. Vu in my supervision team. His expertise strengthens the computer science element in my research. I am thankful for his availabilities to discuss and to give me valuable inputs for my research (despite the critical time difference between Delft and Melbourne). In all, I gratefully appreciate the great support, encouragement and patience from all my supervisors, which were vital at times to keep me moving.

I would like to thank my doctoral committee members for their assessments of my dissertation - A/Prof. Vlahogianni, Prof. Tampère, Prof. Eisemann, Prof. Ahn and Prof. Hoogendoorn. Their constructive comments helped me very much with improving the quality of my dissertation.

Deep gratitudes go to my dear Dittlabers (and some of their partners), whom I spent most of the time with inside or outside our cosy offices. Thanks Panchamy & Jerry, Ding, Boudewijn, Leonie, Sanmay, Zahra, Guopeng, Kristel, Muriel, Peter, Salil, Ali. Thank you for welcoming me (and Huong) to Dittlab, Delft and the Netherlands. I enjoyed working and spending time with all of you very much. It would be a mistake not to include the Dittlab visitors, Loic, Juan, Rafal. Thanks, Loic, for showing us around Paris during that exciting summer. I would like to specially acknowledge Panchamy for so much support that she has given me along the way. Thank you for countless

suggestions, pieces of advice and great friendship. The fact that I would consider you as my (unofficial) supervisor says it all.

Sports, especially football, play a vital role in my life. I quickly became a member of two fantastic clubs, VDFC and DCF Inter. I am truly grateful for all the joys that we have shared on and off the pitch. Exceptionally thank all the VDFC members and their partners, which are essentially the core members of the Vietnamese community in Delft. Cảm ơn anh Được, anh Linh & chị Thu, anh Hùng, anh Hiếu & chị Trang, anh Nghi, anh Sanh & chị Hoàng Anh, anh Phan Anh & Ninh, anh Thảo & chị Thảo, anh Thành, anh Chí, anh Sơn, anh Minh & chị Vinh, anh Tuấn Anh, Tùng & Nhung, Vịnh & Diễm, Thiên, Nhật Anh, Huy, Phương & Hiền, Phương (Sing Sing), Chiển. And of course, big thanks to the joint T&P - DCF members, Alphonse, Alessandro, Florian, Vincent, Pablo, Xavi, Nikola, Martijn, Paul, Peyman, Bahman, Yihong and many others. A special acknowledgement goes to Alphonse with 'ridiculous' football skills. Thank you for all the discussion and lessons, Alphonse. I would also like to thank Simeon for many occasional bouldering sections. This sport was new to me, yet it was so much fun trying out-of-comfort-zone challenges with you, Simeon.

I would like to thank many T&P colleagues Xavi, Alexandra, Bernat, Gulia, Marie-Jette, Niharika, Paul, Jishnu, Freddy, Ilse, Konstanze, Malvika, Yufei, Tim, Victor, and others, for many random discussions, outings, etc. I appreciate the friendly working environment that we have in our department. Outside the office, big thanks to Elena, Max, Sander for their warm welcoming me and Huong. We had so much fun at the Balpol 356.

On May 28, 2020, my wife gave birth to our beautiful son - Minh-Dang. I was full of happiness, but so many worries about taking care of my little family (especially during the Covid-19 pandemic). Fortunately, I received so much parenting advice and care from many great friends: anh Linh & chị Thu, Vịnh & Diễm, Ehab & Shahad, Alessandro & Elisa, Léonie, Simeon, Xavi, Pablo. Thank you for accompanying me on this new but exciting time, and making it much less challenging. Special thanks to Linh's family for their kindness and countless supports.

My deepest gratitude is for my family who has constantly supported me not only for this five-year journey but throughout my entire life. Con cảm ơn ba mẹ, chị bé và bé em đã luôn yêu thương, ủng hộ con/em/anh. Hi vọng luận văn này sẽ là một món quà nhỏ để ba mẹ, chị và em thêm tự hào. Con xin chúc ba mẹ và hai chị em (cùng hai gia đình nhỏ thân yêu) thật nhiều sức khỏe và nhiều niềm vui phía trước.

And last but not least, all the credits have to come to my wife and my little son. She has accompanied me the entire journey and absolutely has made these five years much sweeter than it could ever be without her. The birth of our baby is truly a miracle. He has given me so much strength to keep me steadily moving forward. Hương, anh cảm ơn em nhiều lắm. Cảm ơn em đã đồng hành cùng anh trên con đường đời. Cảm ơn em đã chăm lo cho gia đình mình một cách thật chu đáo trong khi anh mãi vật lộn với những nghiên cứu của mình. Anh chắc rằng luận văn này sẽ không thể hoàn thành nếu

không có em bên cạnh. Anh hi vọng em sẽ xem nó như thành quả của chính em và hãnh diện về nó. Minh Đăng, con trai của ba mẹ. Một ngày nào đó con sẽ đọc được những dòng này. Ba muốn con biết được rằng con là điều tuyệt vời nhất đã đến với ba mẹ. Luận văn này sẽ như là một ví dụ cho bài học đầu tiên ba muốn dạy cho con "Không bao giờ bỏ cuộc con nhé". Ba mẹ yêu thương con nhiều!

Tin Nguyen, July 2021

Contents

1	Introduction	1
1.1	Research motivation	1
1.2	Research objective, questions and scope	3
1.3	Scientific contributions	4
1.4	Practical contribution	5
1.5	Thesis outline	6
2	The CoSI methodological framework	9
2.1	General framework	10
2.2	Patterns collection	11
2.2.1	Context	11
2.2.2	Conceptual framework and requirements	12
2.3	Feature extraction	13
2.4	Pattern annotation	14
2.5	Applications	15
3	Pattern collection	17
3.1	Introduction	18
3.2	Methodology	19
3.2.1	Network partitioning	20
3.2.2	Traffic speed reconstruction	21
3.2.3	Congestion marking	22
3.2.4	Congestion graph generation	23
3.2.5	Congestion clustering	24

3.2.6	Congestion-path tracking	24
3.2.7	Overlapping-path merging	25
3.3	Complexity analysis	27
3.4	Case study	29
3.4.1	The Netherlands highway network	29
3.4.2	Traffic data	29
3.4.3	Computational resource	29
3.4.4	Results	30
3.5	Conclusion	34
4	Feature Extraction (i) - Generic Key-points versus Traffic-related Patterns	37
4.1	Introduction	38
4.2	Literature Review	39
4.3	Methodology	42
4.3.1	Overall framework	42
4.3.2	Point-based feature extraction	45
4.3.3	Area-based feature extraction	48
4.3.4	Synthesis	55
4.4	Evaluation Methodology	55
4.4.1	Dataset	56
4.4.2	Watershed segmentation evaluation	56
4.4.3	Clustering evaluation	57
4.5	Results and Discussion	59
4.5.1	Size of the (visual) word dictionary	59
4.5.2	Watershed segmentation of congested patterns	59
4.5.3	Clustering of congested patterns	61
4.6	Conclusions	70

5	Feature Extraction (ii) - Bottleneck Detection and Characteristics	73
5.1	Introduction	74
5.2	Literature review	75
5.3	Proposed framework	78
5.4	Congestion detection	80
5.4.1	The Chan-Vese model	80
5.4.2	The Chan-Vese model for traffic congestion detection	81
5.5	Bottleneck identification	82
5.5.1	Speed discontinuities detection	82
5.5.2	Activation location and time identification	84
5.5.3	Identification of associated congestion	85
5.5.4	Primary or secondary bottleneck determination	85
5.6	Methodology verification	87
5.6.1	Traffic congestion detection	87
5.6.2	Verification of the bottleneck identification method	90
5.6.3	Time complexity	93
5.7	Case study	94
5.7.1	Detection of bottlenecks on a field-data pattern	95
5.7.2	Derived insights into the selected corridor	95
5.7.3	Time complexity	98
5.8	Conclusion	98
6	Pattern Retrieval	101
6.1	Introduction	102
6.1.1	The necessity of congestion pattern retrieval	102
6.1.2	General retrieval framework	104
6.1.3	Chapter outline	104
6.2	Pattern representation	105
6.2.1	Terminology	105
6.2.2	Feature extraction	106

6.2.3	Relation-graph formulation	108
6.3	Similarity measurement	110
6.3.1	Brief overview of graph matching	110
6.3.2	Phase 1: Nodes similarity	111
6.3.3	Phase 2: Nodes mapping	114
6.4	Experiment results and discussion	115
6.4.1	Data & parameter settings	115
6.4.2	Retrieval results	116
6.4.3	Parameter impacts	120
6.4.4	Time complexity	122
6.4.5	An opportunity for semantic retrieval	123
6.5	Conclusion	124
7	Conclusion	127
7.1	Main conclusions and Key findings	128
7.2	Implications for practice	130
7.3	Recommendations for future research	131
	Appendices	135
A	Effects of ASM parameters on the recognition of traffic disturbances	137
A.1	Smoothing kernel broadness	138
A.2	Propagation speed	138
	Bibliography	141
	Summary	149
	About the author	153
	TRAIL Thesis Series	157

Chapter 1

Introduction

1.1 Research motivation

Mobility is an important part of human daily life, in which car travel is one of the most popular modes. For instance, around 70 per cent of the total number of travel distances in the Netherlands were done by car in recent years (KiM, 2019). The shared proportion in 2017 is approximately 131 billion kilometres, of which the most frequent reasons for travelling are to cover regular aspects of life such as work, social and recreational activities. Therefore, having and maintaining high-quality mobility on highway networks is arguably vital to transportation systems.

One of the main challenges of car traffic is to mitigate congestion which is the state of still or slow-moving vehicles. It strongly impacts human life in various aspects (Falcocchio & Levinson, 2015). From an economic perspective, the direct consequence of traffic jams is increases in travel time; that means motorists waste their time in congestion. Also, research has shown that increased congestion negatively influences economic growth. Particularly, it dampens the growing rate of employment and income (Jin & Rafferty, 2017; Hymel, 2009). Besides, from an environmental point of view, the air quality surrounding the congested-intersections neighbourhood is also degraded due to the high density of vehicle emissions (Hao et al., 2017).

In order to solve or reduce congestion, a better understanding of traffic or more effective traffic management and policy evaluation is needed. The achievement of this would require extensive studies which are necessarily based on the foundation of underlying (sensing) data representing related traffic. Reaching an effective solution for congestion mitigation from data involves various steps. To concisely represent this process, we adopt the so-called DIKW model (Ackoff, 1989; Rowley, 2007) which represents knowledge as illustrated in Fig. 1.1. Collected measurements are at the lowest level, i.e. the *data* level. In fact, increasing amounts of traffic data have been collected on highways by using various sensory systems such as inductive loops, AVI - Automatic Vehicle Identification and FCD - Floating Car Data. By combining these

measurements, such as putting them into a time series or further adding spatial details, the data start to generate more information into traffic. This is categorised as the second layer of the model. The next level aims to gain knowledge, to answer the 'how' questions, based on the information in the lower level. The outcomes are, for instance, mathematical models that represent traffic dynamics or simulation tools based on those models (e.g. (Van Lint et al., 2005; Wang et al., 2006; Calvert et al., 2011; Spiliopoulou et al., 2014)). Such important pieces of knowledge are the stepping-stones for making decisions at the top-most level. Those include policy evaluation (Van Lint et al., 2008; Zheng et al., 2012), traffic management and control (e.g. van de Weg et al. (2018); Soriguera & Robusté (2011); Vlahogianni et al. (2005)), to name just a few applications.

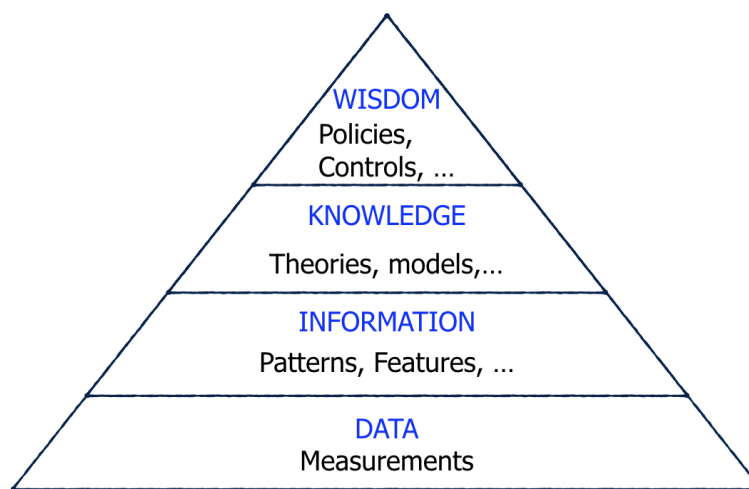


Figure 1.1: The pyramid of knowledge, i.e. the DIKW model

In the *information* layer, spatio-temporal representation of congestion, which shows the status of traffic at multiple locations on a route over a certain time duration, possesses many advantages. From empirical assessment, such a map is very informative as the onset and the dynamic (e.g., the propagation) of congestion can be conveniently observed. Characteristics of congestion (and traffic, in general) like shock wave speed can be measured intuitively by taking into account information in the spatial dimension (e.g. (Schönhof & Helbing, 2007)). Moving on to further studies, congestion patterns can be beneficial to models development. Collecting a wide range of congestion patterns can reveal recurrent instances which might be used for future prediction. That means traffic having the same state to a previously seen pattern potentially leads to similar future states. For instance, expected travel time can be predicted by projecting the current traffic state into the cluster of similar patterns in historical data (Yildirimoglu & Geroliminis, 2013). Therefore, these 2D patterns of congestion are a valuable source of information for various applications in the traffic domain.

Collecting, managing and making use of congestion patterns in the era of big data are challenging tasks. Manual conduct is time-consuming and, therefore, inefficient and most likely impractical. For instance, to validate a traffic model, the developers need to

find relevant examples, e.g., congestion patterns, in data. Most of the time (if not all), this is done using a try-and-error approach which is experience-based, therefore, less effective. Consequently, these challenges call for automatic methods and tools to deal with large traffic datasets. Bridging this gap could improve the effectiveness of various studies in the field of transportation. In particular, having a rich layer of information (in advance) essentially reduces or saves the expensive cost of processing massive datasets at once. It assures that relevant information, for example, related to traffic congestion, are ready to be further analysed by applications at higher layers. Furthermore, it can also promote large-scale studies and probably encourage more thorough evaluations of methods given a better utilisation and/or accessibility of big datasets. That is the main focus of this thesis.

1.2 Research objective, questions and scope

The overall research objective of this thesis is as follows:

To develop efficient algorithms and methodologies to automatically collect and learn congestion patterns from large volumes of traffic data, in order to enrich traffic information.

To achieve this overall goal, key research questions are subsequently formulated as follows.

Research question 1. How can congestion patterns be collected periodically from traffic data in large-scale highway networks?

Traffic congestion reflects the quality of transportation networks although it is not the dominant state of traffic. Thus, identifying and storing occurrence of congestion a priori is beneficial to future retrieval. Congestion can propagate through intersections and spread to upstream links. If network topologies or traffic is complicated enough, multiple intersections might involve in. That means to have a *complete* set of spatio-temporal representations of congestion require dealing with these intersections properly. In addition, traffic data are collected continuously, which means the developed method needs to be computationally efficient to keep up with the incoming data.

Research question 2. What are the representative features of (highway) congestion patterns?

Given, for instance, a nation-wide highway network, hundreds of congestion patterns could be expected in a single day, and thousands are certainly possible over months. Therefore, it is necessary to label them so that further retrieving is efficient. By putting patterns into various categories, which represent different types of congestion, we can reduce the searching space when looking for a target pattern. In order to annotate patterns effectively, representative features need to be engineered carefully. Ideally,

a good set of those infers small distance measures between patterns of similar types, whilst longer distances are expected between those of different types.

Research question 3. How can domain-specific features, namely characteristics of traffic congestion which are well recognized in the literature, be learned automatically?

This question is relevant in case of recognizing the advantages of domain-specific features, which is highly possible, from the previous question. As mentioned previously, thousands or millions of congestion patterns could be obtained over time. Manually organising/annotating such amount of data is cumbersome if not impossible at all. Therefore, automation of the feature extracting method is crucially important.

Research question 4. How can similarities between congestion patterns be ranked using features that are learned previously?

There are possibly various features that can be learned from congestion patterns as the result of answering the previous questions. Subsequently, it is important to combine them in certain ways so that they can be used to measure pattern similarities quantitatively. These quantities are the proxy for matching most similar patterns to a given example. This is relevant to situations in which, given certain traffic congestion situation, we are interested in the most resembling patterns of congestion occurred in the history or in other places (in a network).

The scope of this thesis is to look at spatiotemporal patterns of speeds on uninterrupted motorway facilities. That means we will consider traffic speeds averaged over lanes.

1.3 Scientific contributions

Overall, in this thesis we develop automatic and efficient methodologies to enrich and also boost the accessibility of traffic information from vast amounts of collected data. In particular, we mainly focus on how to automatically identify and characterize traffic congestion on large-scale highway networks. Key scientific findings from this thesis are listed in the following.

1. The development of a method to identify and extract spatio-temporal patterns of congestion from collected measurements of traffic on large-scale highway networks. The method is computationally efficient and could be executed periodically for handling streaming of traffic data (*Question 1*).
2. The development of a method to identify spatial and temporal extents of congestion in resulting 2D patterns. The resultant spatio-temporal boundaries are beneficial to further traffic-related measurements. For example, total delay due to congestion can be accurately measured from data instead of (subjectively) assuming a threshold for congested speed. (*Question 3*)

3. The development of a Watershed-based method to separate (spatio-temporal) a pattern of congestion into various segments which are related to different traffic phenomena like wide moving jams and homogeneous congestion. This basically provides an efficient way to extract traffic-related characteristics of congestion patterns. (*Question 3*)
4. The development of a method to disentangle congestion patterns for the identification of bottleneck activations, especially in complicated situations where multiple close bottlenecks are involved in a pattern. Bottlenecks are one of the main sources of regular traffic congestion. Therefore, this work is highly relevant as it reveals those *hot spots* automatically from traffic data. Long-term statistics can subsequently be derived which essentially provide a rich source of information for measuring the impacts of existing bottlenecks. In addition, by disentangling activations, *primary* and *secondary* bottlenecks are also identified. Such correlation information is beneficial to traffic management and control such as making early (and reasonable) decisions to mitigate traffic congestion in the future. (*Question 3*)
5. Insights into the comparison between generic and domain-specific feature schemes from the perspective of congested traffic patterns. Both of these have their own advantages and disadvantages. By transform speed maps into intensity images, the so-called point-based features tend to group patterns with similar textures, which is potentially one of the valuable features to measure pattern similarities. However, due to the semantic gap (of local features and traffic patterns), it is a challenge to interpret its outcomes. On the other hand, domain-specific features are more parsimonious and better organize existing patterns with respect to traffic knowledge. However, it requires the developments of related methods for extracting those customized features. (*Question 2*)
6. The development of a method to match and rank similar patterns by combining both generic and domain-specific features. The method indicates the advantages of the combination of both feature schemes. In particular, the generic features favour the similarities in the texture of (image) patterns, whilst the graph-based representation of domain-specific features focus on the structure (i.e. the association of different traffic phenomenon) of the presented congestion. (*Question 4*)

1.4 Practical contribution

This thesis designs a system that aims to manage congestion patterns emerging from daily traffic data. It focuses on developing automatic methods and tools that are dedicated to handling large amounts of data. The following lists all practical contributions of this thesis.

1. The development of two methods to automatically detect congestion in large-scale networks. The first method identifies individual clusters (alternatively, junks) of

emerging congestion. The second method determines the spatial and temporal extents of related congestion. To traffic managers or practitioners, consequences of congestion, such as total delay, can be measured efficiently. In addition, the automation feature assists greatly the calculation of either long-term statistics, or large-scale networks. (*Question 1, 4*)

2. The development of a Watershed-based method to segment a spatio-temporal representation of traffic state (namely speed) in congestion. The extents of wide moving jams are effectively and automatically determined. This method can support the calibration of shock wave speed in related traffic models. (*Question 3*)
3. The development of a method to automatically identify bottleneck activations in traffic data. The resulting congestion is also obtained. These are key elements to measure or rank the influences of existing bottlenecks on the performance of a traffic network. Accordingly, relevant strategies can be evaluated based on these measures to mitigate impacts of highway bottlenecks. (*Question 3*)
4. A proof-of-concept of a searching engine on large datasets. Patterns of congested traffic are collected and annotated with representative features which facilitate future retrieval. The system supports various ways of exploration such as using keywords or example patterns. It can significantly and efficiently improve the accessibility of traffic data to practitioners. (*Overall objective*)

1.5 Thesis outline

The structure of this thesis is depicted in Fig. 1.2.

Chapter 2 is dedicated to the conceptual design of the database of congestion patterns developed in this thesis. It is so-called CoSI (**Congestion Search engIne**). CoSI consists of all aspects which are related to the selection and management of spatio-temporal patterns of congested traffic. In this chapter, we present the methodological designs of all required components which are in fact addressed in following chapters.

In Chapter 3 we present a method to identify congestion from a (continuous) streaming of high-volume traffic data, speed measurements in particular. Both spatial and temporal dimensions are incorporated to construct 2D patterns of traffic congestion. Divide-and-conquer strategy and parallel processing are adopted to enhance the feasibility of the proposed approach on large-scale networks.

The next two chapters, Chapter 4 and Chapter 5, are related to feature learning. Ultimately, the objective is to identify representative features of congestion patterns which effectively discern them. These chapters show how image processing techniques can be applied to the field of transportation to extract either generic features or traffic related characteristics. In the former chapter, we compare and analyse the advantages and drawbacks of these features in the clustering and classification applications of

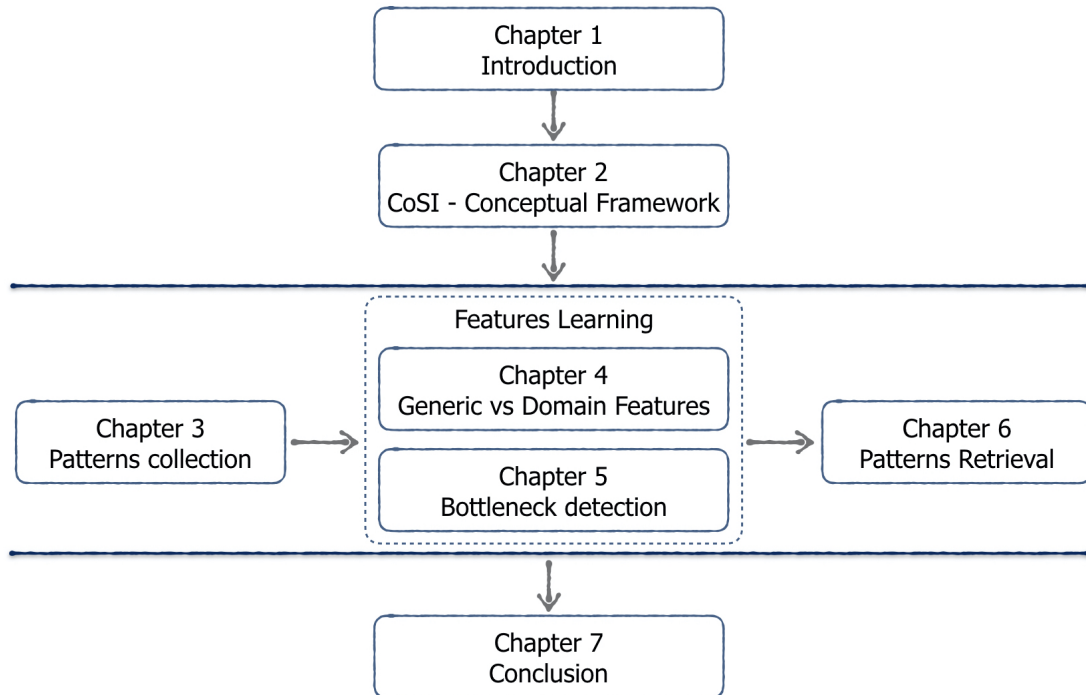


Figure 1.2: Structure of the thesis

traffic patterns. In the latter chapter, we present an automatic method to disentangle patterns for the information about bottleneck activations. The benefits of the method are twofold: it provides an important feature into describing congestion patterns; it enables extracting long-term statistics of traffic bottlenecks which are strongly relevant to traffic management and control.

Chapter 6 presents the retrieving application on the collected patterns of congestion. We focus on pictorial searching which is one of the most challenging problems in image retrieval. Given various features learned in previous chapters, this chapter describes how they are used to represent congestion patterns. Furthermore, matching algorithms which measure the similarity between any pair of patterns based on those representations are proposed. The chapter also analyses and compares these two feature schemes and their combination with respect to the similarity ranking application.

Finally, Chapter 7 presents the conclusions, key findings based on our research. We further discuss the practical implication of the CoSI system. We close the thesis with recommendation on feature research.

Chapter 2

The CoSI methodological framework

The research in this thesis is devoted for the development of an intelligent search engine for patterns of highway traffic congestion. This chapter discusses the conceptual framework for a such system. Fundamental tasks and related components are identified, including (i) digitally traversing a highway traffic network to collect all patterns of congestion that had occurred, (ii) extracting salient characteristics of these patterns, which are critical for (iii) annotating patterns to benefit information retrieval at higher layers. Related functional descriptions and requirements will be discussed which give the foundation for research that will be presented in subsequent chapters.

2.1 General framework

Vast amounts of traffic data have been being collected from sensory devices like inductive loop detectors. These devices measure traffic variables like vehicular speed or flow at a certain point in location and time. Individually, one measurement merely provides an understanding of the ongoing traffic situation. Furthermore, navigating through a great amount of these individual measurements for relevant information is cumbersome. In this chapter, a conceptual framework for more effective utilisation of collected traffic data is proposed as depicted in Fig. 2.1, which is named CoSI.

The CoSI framework comprises different components that conceptually can be categorised into database construction and application. The former is responsible for constructing a database of congestion patterns which can later promote smart retrieval. To this end, the former part is designed with having three layers: (i) patterns collection, (ii) feature extraction and (iii) pattern annotation. Essentially, given collected traffic data, vehicular speeds for specific, the first layer gathers patterns of congestion that have occurred in an entire network. Then, salient characteristics of these patterns are learned by carefully designed machine learning methods. These features are vital for the recognition of differences or similarities between patterns. This is an important layer since it enables smart ways of organizing data and also accessing it. Think about a library, for example, patterns are like books and they should be categorized smartly for efficient and quick access. It is expected that similar books are placed closed to each other. For that purpose, themes and/or topics of a book would be characters or features to start with.

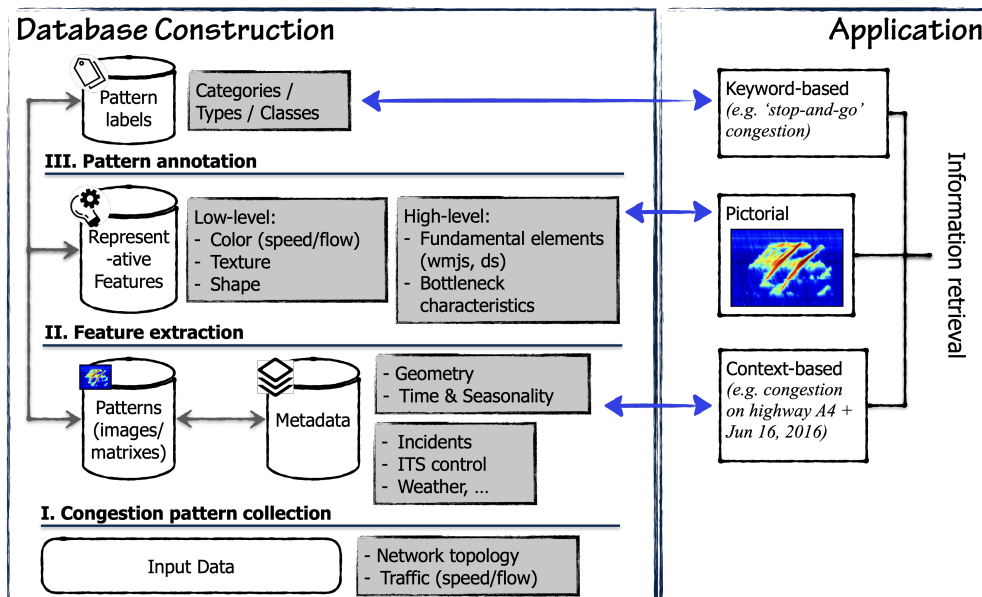


Figure 2.1: General design framework of CoSI

The main application in the CoSI framework is a *smart* search engine, or so-called *information retrieval* engine. Traffic data is collected and mined which results in informative patterns of congestion. This application intends to provide an effective approach

to deliver such information to users as quick and intuitive as possible. Three retrieval options have been identified: contextual, pictorial and keyword-based. Details of these layers/components are provided in the following sections. Furthermore, the connections between the conceptual design of CoSI and the research questions presented in Chapter 1 are also be made clear when applicable.

2.2 Patterns collection

This section discusses the first layer of the pattern database construction, in which spatio-temporal congestion patterns are collected from traffic networks. The design of this pattern collection module is in line with the first research question (*Research question 1.*).

2.2.1 Context

To better manage and control traffic networks, numerous amounts of sensor devices, such as inductive loop detectors, have been implemented. They periodically measure and record traffic information like speed, flow or travel time, etc. Individually, they present traffic dynamics at local locations themselves which is limited as traffic is not just temporal, but also spatial. Hence, there is a need to present traffic both spatially and temporally so that better insights into traffic dynamics can be observed effectively. This is the motivation of having spatio-temporal traffic patterns, of course, the focus is on congestion.

A highway network contains a lot of intersections where congestion can occur at either outgoing links or incoming links. To represent traffic in these situations using 2D patterns requires two things: (i) a method to detect congested intersections and related links, i.e. links that are congested, and (ii) extract all individual routes that congestion can propagate through these intersections. As a result, there is one pattern representing congestion in each route.

Traffic sensors, like loop-detectors, are implemented at sparse locations, hence they only provide information locally. One link can have one or multiple detectors, but there is a possibility that no detector is implemented. Therefore, to better understand the traffic on a network, or a route, for example, one would need to incorporate various sensors at different locations and apply traffic estimations to further derive information at which no sensor is implemented.

Traffic data are being collected continuously, hence congestion patterns need to be collected regularly. This is a repetitive process that requires efficient computation. If a method takes more than one day to process one day of traffic data, it is obviously not sufficient and should not be applied. Note that the pattern collecting module is just the first step in building an intuitive database and there are other modules for further

processing the collected patterns. Therefore, computation is an important aspect of developing corresponding methodologies.

2.2.2 Conceptual framework and requirements

Based on the context described above, we proposed a framework for collecting congestion patterns in a traffic highway network as shown in Fig. 2.2. It consists of three main modules, namely congestion detection, congestion clustering and pattern extraction. In the first module, the network topology is cut into non-overlapping routes so that the corresponding data can be retrieved and assimilated. Congestion is detected upon granularly estimated data. This results in a so-called indicator map of congestion on each route. The second module uses information from network topology to combine those congestion indicating maps. It further clusters existing congestion into different spatiotemporal groups. Note that a (spatiotemporal) region of congestion on a link at a period is either connected to that of the neighbour links at the same period or the same link at the preceding or succeeding period. Finally, the third module processes each cluster separately. It finds all possible routes therein and then uses related traffic data to construct the corresponding patterns of congestion. Details of the development of methodologies for these modules are presented in Chapter 3.

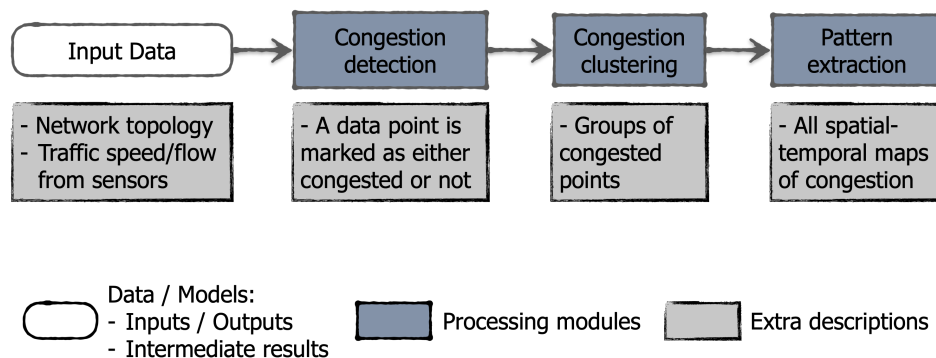


Figure 2.2: Conceptual framework for *Pattern collection*

Regarding computation, a *Split-Merge* strategy is designed to deal with large highway networks, e.g. nationwide. This approach has two advantages. Firstly, processing small networks is less memory-demanded. Secondly, by having multiple *independent* sub-networks, a parallel mechanism can be adopted that potentially speeds up the whole process. Fig. 2.3 illustrates the proposed methodology. The split stage divides a large network into smaller sub-networks. Then, each one of them will be processed independently to extract congested routes by the procedure shown in Fig. 2.2. Depending on the availability of computational resources, multiple sub-networks can be processed in parallel for fast computing. Next, in the merge stage, these congested routes from different sub-networks are examined and stitched together to form complete routes. Note that both spatial and temporal constraints are taken into account. Finally, congestion patterns are constructed from related traffic data from those final routes.

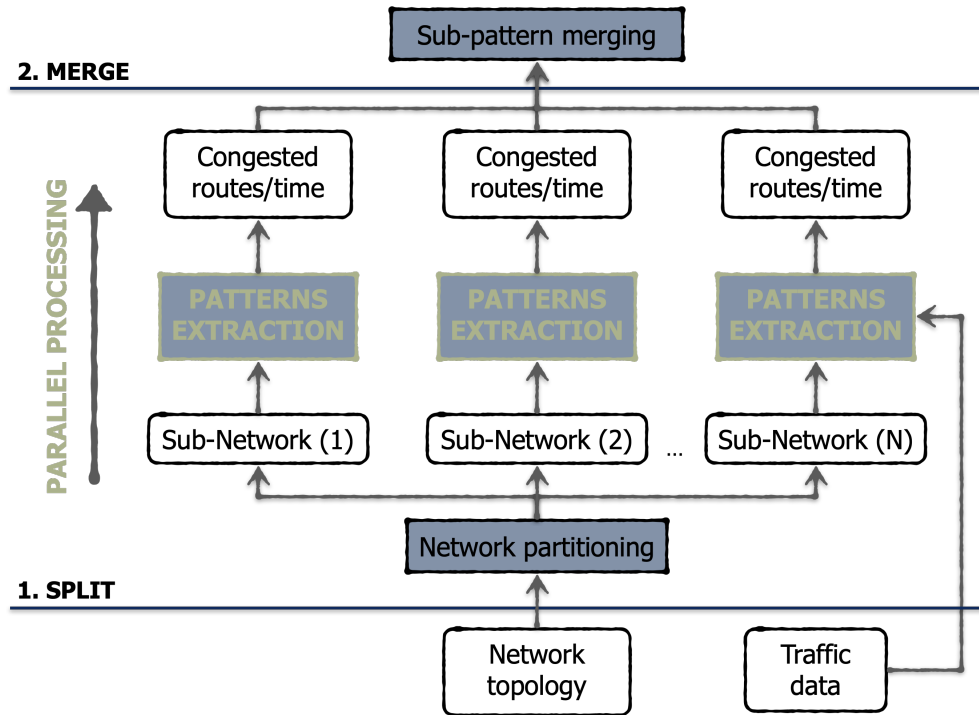


Figure 2.3: Parallel mechanism for the *Pattern collection*

2.3 Feature extraction

The previous section shows how all patterns of congestion are collected from loop-detector data from large-scale networks. In this section, two different features schemes for extracting representative characteristics in each pattern are discussed. Feature extraction plays an important role in implementing a *smart, intuitive* dataset. The design of this feature extraction module is related to the second and third research question (*Research question 2.*, *Research question 3.*). The objective is to differentiate (or detect similar) patterns by identifying unique or discriminative features in each one. Consequently, searching through the database can be conducted accurately based on these features. Besides, insights into various collected patterns of congestion can be learned, e.g. common types of congestion.

Generally, there are two approaches for feature extraction which are generic and domain-specific. The former approach looks at (speed) patterns as intensity images and applies well-developed techniques from computer vision to extract features. The speed (image) maps are widely used by traffic researchers to visualize congestion both spatially and temporally. Hence, regarding them as images is a natural or reasonable approach (In essence, they are essentially 2D matrices of numeric values). Note that, features found by this approach do not provide direct traffic meaning. They are rather related to general features like image gradients, textures, etc. On the other hand, the latter extracting approach aims to identify various features in patterns that are understood or explainable by using traffic knowledge. In this thesis, we focus on the two most observed features in congestion patterns which are the wide moving jam and activated bottlenecks.

For validation, we further compare these two approaches on their abilities to learn representative features. Consequently, two applications that can be of direct use are clustering and classification. The former indicates how well these features can separate different patterns while grouping similar patterns simultaneously. The latter shows how accurately a pattern is placed into one of the predefined groups. These two machine learning processes are significantly relevant to the next module, which is pattern annotation.

The necessary methodologies for this feature extraction module are presented in Chapter 4 and Chapter 5.

2.4 Pattern annotation

Given a collection, i.e. a database, of many congestion patterns, it is important to annotate or associate each pattern with relevant labels. This is an important action to achieve efficient information retrieval to the database. In this section, a framework is conceptually designed for labelling congestion patterns, which includes two important aspects. Firstly, there are many patterns of congestion that will be collected, therefore, automatic labelling is required since a manual approach is arguably infeasible. Secondly, as congestion patterns are continuously collected, the adopted scheme needs to cope with such new arrivals.

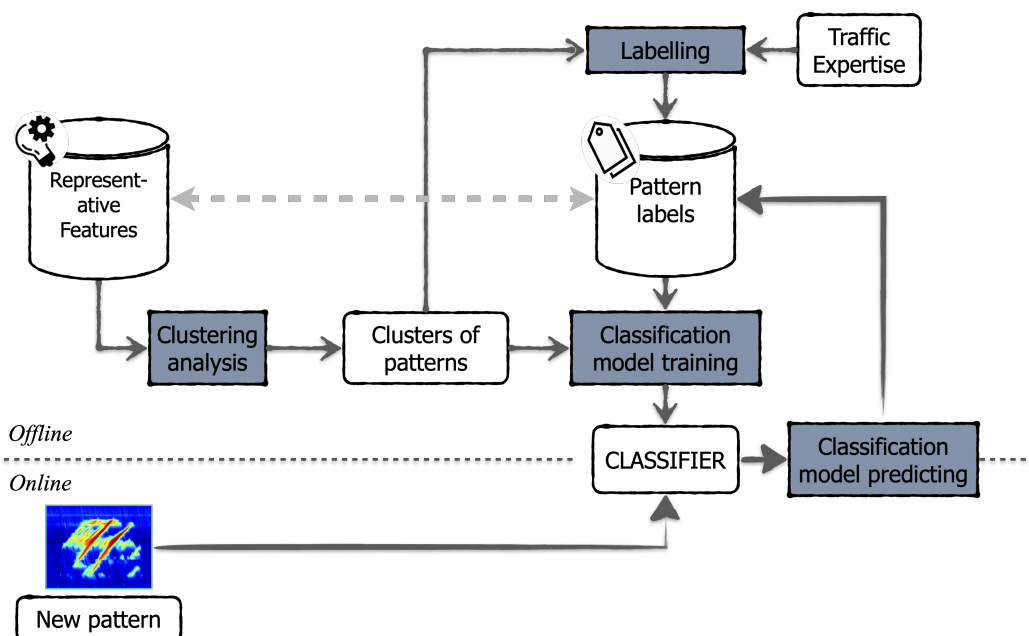


Figure 2.4: Conceptual framework for *pattern annotation*

The proposed framework consists of two phases, i.e. offline (learning) and online (applying), as shown in Fig. 2.4. In the offline learning phase, patterns are clustered and assigned names (class names) by traffic experts. Since the number of patterns is

significantly high, a manual approach would be labour intensive. Therefore, clustering analysis is performed on features that were learned in the previous layer. Then, experts can simply examine and give labels to groups of patterns. This approach is much more efficient. In the online phase, we also want to label newly collected patterns. Again, to automate the process, a classifier is trained on the clusters learned during the offline phase. This classifier is then applied to figure out labels for new patterns.

Although it has not been addressed in this thesis, it is worth mentioning the challenge that a new class of patterns evolves. For that, two main components are required, namely the outlier detection and classifier update. The former is responsible for identifying the emergence of a new type of congestion pattern. Theoretically, it is related to the sensitivity of the classifier and other possibly relevant parameters of the definition of the existing clusters. The latter is concerned with how to update the current classifier to cope with a new class.

2.5 Applications

This section discusses the ultimate goal of the CoSI framework, which is to support applications that use annotated patterns. Examples include case-based decision support systems and applications related to traffic estimation and prediction. In this thesis, the focus is on data retrieval and more specifically efficiently searching through large traffic databases. This application emerges naturally as a number of patterns representing congested traffic can probably be collected and stored by realisations of the database's design in previous sections. As shown on the right-hand side of Fig. 2.1, the proposed framework promotes three options for the retrieving task, namely context-based, keyword-based, and pictorial searching.

The first retrieval option finds patterns given contextual information such as where (specific locations) and/or when (time) that traffic is congested. This is the most basic, direct and simplest way to utilise the database. By successfully achieving the first layer of constructing the database, i.e. pattern collection part, incorporating with metadata (location/topology/time), this retrieval option can be completely fulfilled.

While the first option only relates to the context of traffic congestion, the remaining two options concern the content therein. In particular, the second option finds patterns by labels or annotations, which are the result of the learning done on collected patterns in the database. Thus, this so-called keyword-based retrieval is related to the third part - pattern annotation - of the database construction. Available keywords are subsequently obtained by empirical studies on congestion patterns. In this framework, we adopt different types of congestion as possible keywords for labelling and retrieving.

The third option is the so-called pictorial searching in which an example of expected patterns of congestion is provided in advance, then the system looks for similar patterns in the dataset. Developing this module also provides an answer to the fourth research

question (*Research question 4.*). This is the most sophisticated retrieval method among the available options since it requires the extraction of the example's characteristics. Hence, naturally, this part is linked to the second part of the database construction, i.e. feature extraction. There are two possible deliveries of the outcomes at this stage. The first one, which is similar to the keyword-based option, determines which category an example pattern belongs to and retrieves some (arbitrary) patterns from that group. The second approach is more thorough by locating patterns (probably in the identified category as in the first approach) that share common characteristics as presented in the given example.

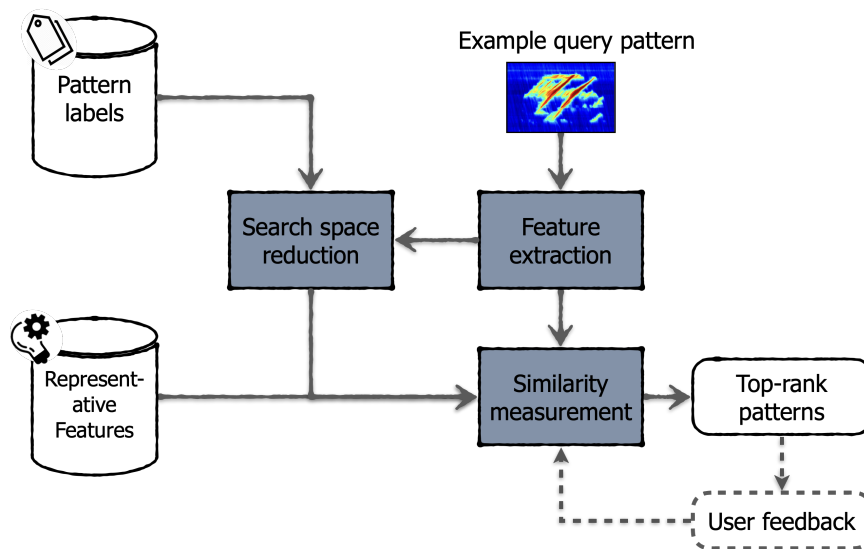


Figure 2.5: Conceptual framework for *pictorial retrieval*

It can be seen that the last option is the most challenging functionality of the application. Fig. 2.5 shows a general framework for similar pattern searching. The feature extraction component is developed during the implementation of the second layer of the database construction design. It identifies important characteristics in the given pattern in the same regime as when processing other patterns in the database. The similarity measurement component compares the features in the given pattern with features of those patterns in the database. Then, the most similar patterns can be identified. The detailed study of this pictorial retrieval is presented in Chapter 6. It is worth noticing that the number of available patterns accumulates daily. Broadly discovering similar patterns in the whole database might be impractical even though that depends greatly on the approach. Therefore, a suitable scheme for reducing the search space is necessary. That way, only a limited number of patterns are considered, hence, processing time can be shortened, i.e. quicker response.

Chapter 3

Pattern collection

Traffic dynamics in general and congestion in specific are better observed in spatio-temporal dimension. Hence, having these 2D representations of congested traffic a priori offers a convenient approach to explore traffic through sensory data. In a road network, congestion can emerge at any location and propagate upstream through various intersections and involve various roads. This chapter presents a framework for identifying congestion in a traffic network and constructing 2D patterns of derived congested traffic. The underlying conceptual framework has been given in Chapter 2. In addition, specific modules and related methodologies are proposed and discussed in detail. *The source code of the developed methodologies in this chapter is publicly accessible at this repository: https://github.com/nguyenthientin/pattern_collection*

3.1 Introduction

At the corridor level, the dynamics of traffic are better observed by using spatiotemporal maps. Relevant quantities such as vehicular speeds or flows can be visualised over both the spatial (i.e. over a path) and temporal dimensions. These status maps potentially benefit different applications or studies. For instance, the characteristics of traffic congestion are visually and intuitively illustrated. Accordingly, congestion shock waves can be observed, and their related speeds are efficiently measured (e.g. Schönhof & Helbing (2007)). Furthermore, collecting a wide range of these 2D maps potentially reveals recurrent patterns of congestion that can benefit traffic state prediction. For instance, an ongoing traffic state can be classified into a group with similar patterns in historical data. Then experienced travel times from this cluster are used as predictors for the expected travel time (Yildirimoglu & Geroliminis, 2013). We refer to these 2D maps as spatiotemporal (or 2D) congestion patterns onward.

Once triggered in a network, congestion potentially propagates upstream and passes through various intersections. This situation most likely happens during peak hours when the travel demand increases significantly and surpasses the capacity of the underlying road section. Obtaining 2D congestion patterns is not a trivial task in those complicated scenarios, especially when many intersections involve. Exhaustively searching for all congestion patterns on all possible routes is not a feasible approach since the possibilities grow exponentially with respect to the size (or the complication) of the network. Hence, this calls for a method that automatically collects spatiotemporal congestion patterns from traffic data on large-scale highway networks. The problem is formulated in Definition 3.1. Some conditions are declared to assure a complete, non-redundant set of congestion patterns.

Definition 3.1 *Given a (highway) traffic network and related data over a certain period, we propose that a method that collects existing patterns of congestion needs to meet the following criteria.*

- C1. (**Pattern definition**) *Each pattern is a spatiotemporal map of traffic states representing congestion that occurred on a path over a period.*
- C2. (**Pattern completeness**) *A pattern represents the entire propagation of related congestion both spatially and temporally.*
- C3. (**Collection completeness**) *A collection of patterns covers all instances (or spatiotemporal regions) of congestion in the entire traffic network.*

An example is illustrated in Fig. 3.1, which shows a toy network with two intersections connecting five links. At the time instant t_0 , congestion involves three links e_0 , e_2 , e_4 , while at the next time instant t_1 , link e_3 becomes congested. Following the Definition 3.1, two patterns are expected from a pattern collection method: P_0 which includes $\{(e_4, t_0), (e_2, t_0), (e_0, t_0), (e_4, t_1), (e_2, t_1), (e_0, t_1)\}$; and P_1 which includes $\{(e_2,$

t_0), (e_0, t_0) , (e_3, t_1) , (e_2, t_1) , (e_0, t_1) . These two patterns P_0 and P_1 essentially track congestion on two paths p_0 , p_1 (see Fig. 3.1), respectively. Both P_0 and P_1 satisfy the condition C1 and C2. In addition, the collection $\{P_0, P_1\}$ covers all the congested regions in the network. An example of a violation of C2 is the pattern P on the path (e_4, e_2, e_1) . This P does not trace the origin of congestion on e_4, e_2 . Moreover, P is entirely covered by P_0 (with respect to congested regions); hence, it makes sense to have P_0 instead of P in the final pattern collection.

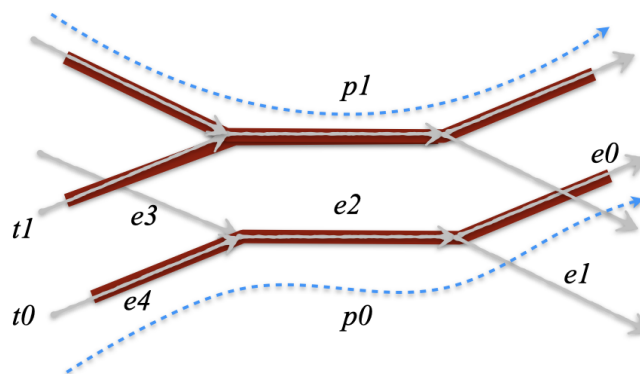


Figure 3.1: A toy example of a traffic network that consists of five links e_0 , e_1 , e_2 , e_3 , e_4 . The arrows indicate the driving direction. Two instances of the network are associated with the time instants t_0 , t_1 . The dark highlighted regions indicate congestion.

The rest of the chapter is structured as follows. Section 3.2 represents the proposed framework and related methods for collecting congestion patterns from traffic data. The complexities of the proposed methods are analysed in Section 3.3. In Section 3.4, a case study is used to demonstrate the proposed method. Finally, Section 3.5 concludes this chapter.

3.2 Methodology

This section describes the proposed framework for identifying and collecting spatio-temporal patterns of congestion that occurs in a highway network. It uses two fundamental inputs which are the topological information of the network and the related traffic data (in particular, vehicular speeds). The framework comprises 3 main phases which include the modules: network partitioning, data retrieval, traffic speed reconstruction, congestion detection, congestion clustering, congestion-path tracking and overlapping-paths merging. Fig. 3.2 illustrates the whole process. Details of all components are discussed in the following sections.

To concisely represent all proposed algorithms in this study, a mathematical formulation of a traffic network topology is provided in Definition 3.2.

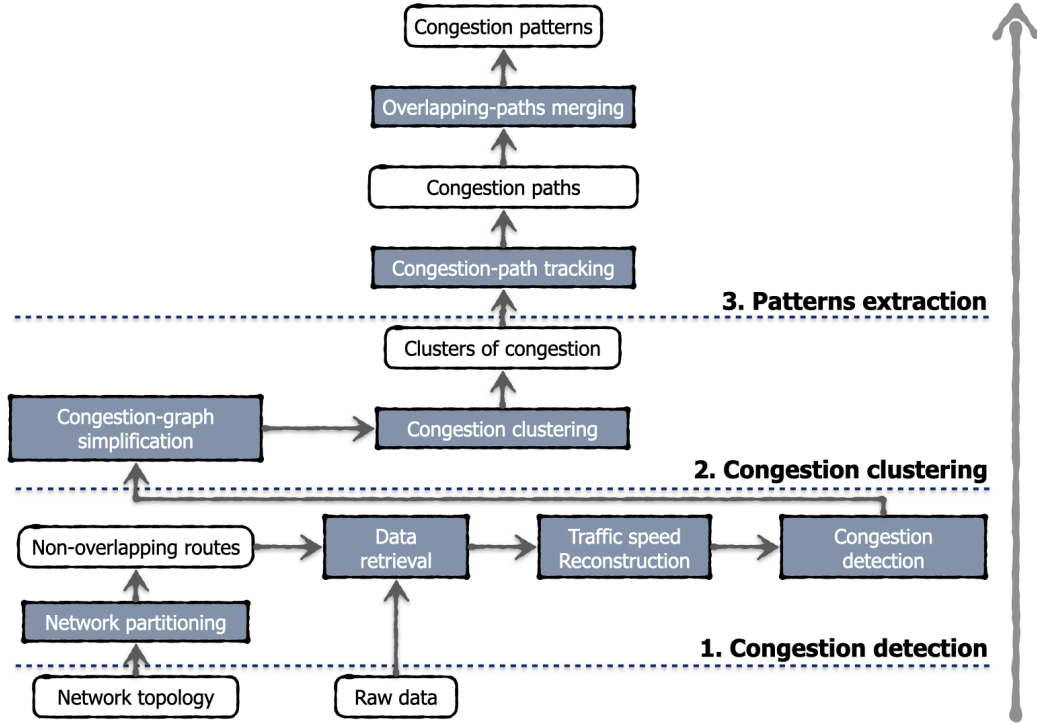


Figure 3.2: The proposed framework for spatio-temporal pattern collection in traffic networks

Definition 3.2 *The topology of a highway traffic network is modelled as a directed graph $G = (E, V)$, where the node set V and the edge set E represent all intersections and links (or segments) in the network, respectively.*

$$\begin{aligned}
 V &= \{v_1, v_2, \dots, v_N\} \\
 E &= \{(v_i, v_j) | v_i, v_j \in V^2, \text{ and there is a direct (travelling) link } v_i \rightarrow v_j\}
 \end{aligned}
 \tag{3.1}$$

3.2.1 Network partitioning

Measurements from sensors, like loop-detectors, represent traffic at discrete locations. To get a better or more complete representation of traffic on a network, it is necessary to incorporate data from various sensors, particularly based on (travelled) paths (or routes). The network partitioning module divides a traffic network into a set of paths, from which traffic states can be reconstructed (or estimated) for locations without sensory devices.

This chapter proposes a naive method that partitions a network for a set of non-overlapping routes. It mostly depends on the topology of the network and related (physical) properties regardless of resulting traffic information. Having said that, different heuristics can be defined to maximize the utilisation of sensory devices and subsequently effectively reconstruct (or estimate) traffic information on locations without sensors.

In essence, the proposed partitioning method is a depth-first search algorithm (Tarjan, 1972). At each node, the order of selecting the next node, and subsequently the next link, is governed by a heuristic rule. In particular, links representing main (or primary) roadways are preferable to those representing secondary roadways. A secondary road is for leaving the road network (off-ramps) or later connecting to another road in the network (turning roads). Topology-wise, characteristics of a main road/link (such as the number of lanes) is more similar to that of the downstream main road/link than the turning road (if available). Therefore, traffic states could be estimated more accurately on the path connecting these main roads, given the presence of detectors thereon. Accordingly, origin nodes on main roads are selected to process first compared to those on connector roads like on-ramps. This heuristic rule allows keeping multiple links of the same main road together.

An example is illustrated in Fig. 3.3. Out of the three paths shown in the figure, r_0 is preferred the most. Firstly, it starts from an origin node that stays on a primary road; therefore, r_0 is preferable to r_2 , which starts with an on-ramp. Secondly, while links of r_0 belong to the same main road, r_1 connects two different main roads (via the connector road). Our proposed heuristics prioritize r_0 over r_2 .

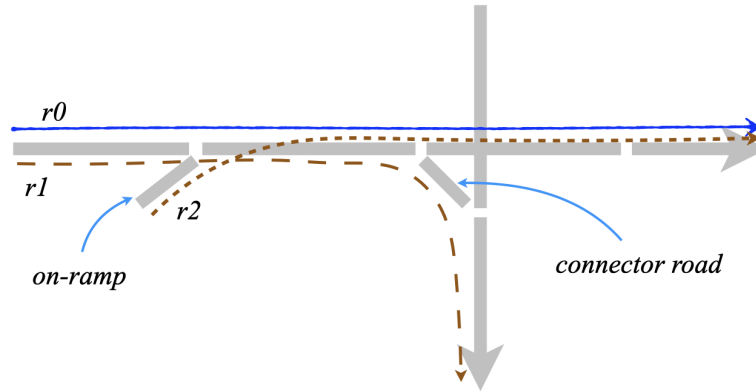


Figure 3.3: An illustration of adopted heuristic rules

The resulted set of paths is formulated in Equation 3.2.

$$\begin{aligned}
 R &= \{r_0, r_1, \dots, r_K\} \\
 r_i &= (v_{m_0}, v_{m_1}, \dots, v_{m_{N_i}})
 \end{aligned}
 \tag{3.2}$$

where, N_i is the number of nodes in path r_i , $v_{m_j} \in V, 0 \leq m_j \leq N, (v_{m_i}, v_{m_{i+1}}) \in E$, and K is the number of paths in the collection R .

3.2.2 Traffic speed reconstruction

Measuring devices, like loop-detectors, are implemented on highways, their spacing differs from few hundreds to thousands of meters. To reconstruct spatiotemporal

speeds at a finer resolution (both spatially and temporally), traffic network is cut into multiple paths by the module described in the previous section. In this module, a filter is adopted to interpolate traffic speeds at (finely equidistant) locations in each of the obtained paths.

For our purposes, we average speeds over carriageways (see Section 1.2 for the research scope), and apply the well-known Adaptive Smoothing Method (ASM) (Treiber & Helbing, 2002; Van Lint & Hoogendoorn, 2010) as the main data assimilation tool. In essence, it is a filter that takes two well-observable traffic phenomena into account: (i) free traffic perturbations propagate in the direction of traffic flow, i.e. driving direction, and (ii) congested traffic propagates upstream in the opposite direction of driving with speeds of approximately -18 km/h (the negative sign indicates moving against traffic direction). ASM offers two advantages in working with loop-detector data. First, it interpolates traffic variables (i.e. speed, in our application) at locations between detectors or recording time instants. Second, ASM also reduces the effects of noises in the collected data. ASM results in more complete *pictures* of traffic dynamics which are significantly valuable for further studies.

3.2.3 Congestion marking

To collect congestion patterns on a road network, we propose to firstly identify congestion on individual paths in R . Then, the topology of the network can be incorporated to associate different parts of congestion to construct corresponding patterns. This section presents a method for detecting congested traffic in spatiotemporal patterns.

As mentioned previously, ASM-filtered data give a better representation of traffic speeds. Hence, it is used as the input data for identifying traffic congestion. In addition, it can be expected that traffic congestion can be better detected in spatiotemporal maps than in individual data points. Besides, the aforementioned state estimation of traffic on a road network has already resulted in a set of paths with (ASM-filtered) data. Therefore, it is logical to identify congestion on these paths. This chapter proposes a (simple) detection method of congestion as follows.

1. Initiating: Define two *relaxed* thresholds for congested U_c and free U_f traffic speeds. These are responsible for an initial filter. Basically, U_f is used to leave out all (spatio-temporal) parts (or data points) that represent high speed traffic since they are not of our interest. The interpretation of U_c is that traffic jams in which vehicular speeds do not drop down below that threshold are disregarded. An option for these parameters are $U_c = 60$ km/h, $U_f = 85$ km/h. Note that more sophisticated methods like percentiles over long periods could reflect local traffic data better.
2. Congestion marking: Mark all speeds below the congested speed threshold as congested. This yields an initial congestion mask $M_c^i \leftarrow u < U_c$.

3. Free flow marking: Mark all speeds higher than the free speed threshold as free. We get the free speed mask $M_f^i \leftarrow u > U_f$.
4. Congestion region growing: Expand the congested region (represented by M_c^i) both spatially and temporally (to get M_c) using dilation operator in image processing (Haralick et al., 1987). As compared to setting another higher congested speed threshold, this approach has two advantages. Firstly, we do not have to choose a single value for the threshold which can vary amongst different patterns (of congestion). Secondly, this approach assures that the congestion of interest causes traffic speed to drop below a certain speed, which is specified by U_c . Equivalently, it serves as a filter to eliminate minor or noise-created congestion. On the other hand, we need to specify how far the M_c^i needs to extend. One can expect sharp changes (either decreases or increases) in traffic speeds at its head and tail. Based on our experiment, we select 2 km and 15 minutes for the parameters used in the dilation operator.
5. Strimming: Due to possible over-expansions in the previous step, the M_c can include free flow areas. To reduce the impact of this miss-detection, we use the free flow mask M_f^i to remove high speeds points from being falsely masked by M_c . In other words, M_f^i acts as the boundary of the expansion in the previous step. The mask after strimming is the final congestion mask.

3.2.4 Congestion graph generation

The previous sections describe the first phase - *congestion detection* - of the proposed framework, which results in the mask M_c of traffic congestion in ASM-data. Since this data represents traffic in fine granularities, a great amount needs to be processed. To reduce the complexity of the problem, this section describes a method for an efficient representation of traffic jams in a road network.

The idea is to magnify any positive indicator of congestion (at one location and time instance) to the related link. That means, at a time instance, a link is considered congested if there is congestion at any location on it. This augmentation keeps the complexity of further components at the scale of the original network. Besides, since regions of congestion from different paths need to be associated later, a so-called *congestion graph* is employed to represent congestion in a traffic network. In principle, this graph is directly related to the network topology and attributed with the time dimension by duplicating nodes for different time instances. Subsequently, each congested link is translated into two connected nodes in the congestion graph. These nodes are annotated with the identifications of corresponding nodes in the network topology. As a result, congestion regions from connected (or intersected) paths could be linked by their common nodes.

The congestion graph potentially consists of numerous nodes and links depending on how severe the related congestion is in terms of spatial or temporal extents. It can be simplified by combining congested nodes that are temporally continuous and forming

new super nodes. A time interval representing all the time instances of individual nodes is stored as an attribute of the related super node. All incoming and outgoing links of individual nodes are preserved on the corresponding super node.

3.2.5 Congestion clustering

Imagine the evolution of network traffic state being presented as a stack in which each layer presents the status at a time interval. By looking into this stack, one can observe congestion occurring at different locations and different time intervals. Moreover, these instances of congestion can form clusters of congested, connected nodes and links at sequential time stamps. Both the upstream and downstream fronts of any (spatial-temporal) instance of congestion is bounded by the associated cluster. Hence, congestion patterns from different clusters are independent of one another. This (congestion clustering) module aims to identify those clusters so that they can be independently processed further.

Each cluster of congestion is (by definition) a weakly connected component as all nodes are connected to each other by some path, given directions of edges are ignored. Hence, a simple yet efficient solution is to first clone the graph while losing all directions, i.e. having an undirected projection of the congested graph. Then, a network traversing algorithm like depth-first search (Tarjan, 1972) is applied to partition nodes and the related links to different connected components, i.e. clusters.

3.2.6 Congestion-path tracking

Given a (weakly) connected graph representing a (spatial-temporal) cluster of congestion, this module is devoted to tracking all (spatiotemporal) propagations of congestion therein. The expected outcome is a set of sub-graphs with each one representing the evolution of an instance of congestion both temporally (continuous intervals) and spatially (links of a particular route).

Three requirements are defined for each sub-graph: (i) is a weakly connected component, (ii) consists of nodes and links that project into only a single path, (iii) is the largest, meaning it is not covered by any other sub-graphs that satisfy both (i) and (ii). The first two conditions guarantee that each sub-graph is equivalent to a 2D (spatio-temporal) pattern of congestion. The last condition, in combination with the first two, assures that each sub-graph represents a complete trace of congestion both in the spatial and temporal dimensions.

Algorithm 3.1 describes the proposed algorithm which operates recursively. It is inspired by the depth first search algorithm (Tarjan, 1972). The spatial network (G_x^c) is utilized to keep track of the underlying path. The idea is to keep track of possible paths and related (congestion) graph components at each node.

Algorithm 3.1 Congestion graph traversal for path-based sub-graphs

```

1: procedure TRAVERSE_NODE( $s, G_x^c, G, M, isVisted$ )
    $G^{wc}$ : a (simplified) weakly connected component of the congestion graph
    $G_x^c$ : the underlying spatial network (with respect to  $G^c$ )
    $s = (v_i, t_m)$ : the currently processed node
    $M$ : traversing outcomes (if available) of nodes in the network
    $isVisited$ : the indicator if a node had already traversed
2:   if  $isVisited(s)$  then
3:     return ▷ stop here since the node had already been processed
4:   end if
   ▷ Scan spatially
5:    $S \leftarrow Successor(G^{wc}, s)$  ▷ get all successor nodes of  $s$ 
6:    $S_x \leftarrow Successor(G_x^c, v_i)$  ▷ get all (spatially) possible successor nodes of  $s$ 
7:    $R \leftarrow \emptyset$ : List of paths
8:    $P \leftarrow \emptyset$ : List of sub-graphs
9:   for each  $v_j \in S_x$  do
10:     $S_j \leftarrow \{(v_j, *) \in S\}$ 
11:    if  $S_j \neq \emptyset$  then
12:      for each  $s_j \in S_j$  do
13:        TRAVERSE_NODE( $s_j, G^{wc}, G_x^c, M, isVisted$ )
14:         $R \leftarrow R \cup M(s_j)(R)$  ▷ add routes from  $s_j$ 
15:         $P \leftarrow P \cup M(s_j)(P)$  ▷ add graph components from  $s_j$ 
16:      end for
17:    else
18:       $R \leftarrow R \cup empty\_route$ 
19:       $P \leftarrow P \cup empty\_graph$ 
20:    end if
21:  end for
22:  Add vertex  $v_i$  to all routes in  $R$ 
23:  Add all nodes in  $T$  to all graphs in  $P$ 
24:   $M(s_i)(R) \leftarrow R, \forall s_i \in \{T, s\}$ 
25:   $M(s_i)(P) \leftarrow P, \forall s_i \in \{T, s\}$ 
26: end procedure

```

3.2.7 Overlapping-path merging

The previous module tracks the propagation of any origin (spatio-temporal) nodes. If there is congestion on a (spatial) node is triggered at different instants and the related congested regions only join at downstream locations, i.e. nodes, more than two sub-graphs will have resulted. This leads to redundancy in the final set of patterns. To avoid that, as well as to obtain prime patterns, the overlapping-path-merging module is developed to combine such paths (accordingly, the related sub-graphs).

The most trivial procedure is described as part of Algorithm 3.2, in which mutual

relations between paths are checked. If two paths overlap, they are combined as well as the related sub-graphs. The shorter path is retained but in a different group, as components for further comparisons. The longer path as well as its related sub-graph are updated with the joined path and sub-graph.

Algorithm 3.2 Complete congested path and related patterns extraction

Require:

The congestion graphs G^c

The underlying spatial network G_x^c

- 1: $SS \leftarrow$ Get all source nodes
 - 2: $M \leftarrow \emptyset$
 - 3: $isVisited(s) \leftarrow False, \forall s \in G$
 - 4: **for each** source node $s = (v_i, t_m) \in SS$ **do**
 - 5: TRAVERSE_NODE($s, G^c, G_x^c, M, isVisited$) ▷ traverse this node
 - 6: **end for**
 ▷ Combine paths, sub-graphs
 - 7: P_a : List of all (overlapping) paths in M
 - 8: G_a^s : List of all corresponding sub-graphs in M
 - 9: $(P_s, G_s^s) \leftarrow (\emptyset, \emptyset)$: List of shorter paths and their related sub-graphs, initiated with empty sets.
 - 10: **for each** path, sub-graph $(p_i, g_i) \in (P_a, G_a^s)$ **do**
 - 11: **for each** path, sub-graph $(p_j, g_j) \in (P_s, G_s^s)$ **do**
 - 12: **if** $p_j \subseteq p_i$ **then**
 - 13: Update $p_i \leftarrow p_i \cup p_j$
 - 14: **end if**
 - 15: **end for**
 - 16: **for each** path, sub-graph $(p_j, g_j) \in (P_a, G_a^s)$ **do**
 - 17: **if** $p_j \subseteq p_i$ **then**
 - 18: $p_i \leftarrow p_i \cup p_j$
 - 19: $(P_s, G_s^s) \leftarrow (P_s, G_s^s) \cup (p_j, g_j)$ ▷ move (p_j, g_j) to set (P_s, G_s^s)
 - 20: $(P_a, G_a^s) \leftarrow (P_a, G_a^s) \setminus (p_j, g_j)$ ▷ remove (p_j, g_j) from (P_a, G_a^s)
 - 21: **else if** $p_i \subseteq p_j$ **then**
 - 22: Update $p_j \leftarrow p_j \cup p_i$
 - 23: $(P_s, G_s^s) \leftarrow (P_s, G_s^s) \cup (p_i, g_i)$ ▷ move (p_i, g_i) to set (P_s, G_s^s)
 - 24: $(P_a, G_a^s) \leftarrow (P_a, G_a^s) \setminus (p_i, g_i)$ ▷ remove (p_i, g_i) from (P_a, G_a^s)
 - 25: **end if**
 - 26: **end for**
 - 27: **end for**
-

3.3 Complexity analysis

This section discusses the complexities of various components in the proposed framework. By doing this, the feasibility of applying them to large-scale networks can be justified analytically.

Network partitioning As the network partitioning module only requires topological information, it can be conducted in advance. In addition, the outcome can be reused as long as the network remains the same. Hence, its complexity might not be a major concern. Nevertheless, a simple algorithm with a path growing approach would have a complexity of $\mathcal{O}(|E|)$, in which $|E|$ is the number of edges in the graph G .

Traffic speed reconstruction In (Schreiter et al., 2010a), the authors have provided an in-depth analysis of the ASM method with respect to different algorithms. Table 3.1 provides a summary of the results, which suggest that the complexity of the ASM method is linear in the size (total length X) of a network. Also, the authors have proven that the method is scalable to large-scale networks.

Table 3.1: The complexity of algorithms implementing the ASM method, reproduced from (Schreiter et al., 2010a). Where X is the total length of road segments, T is the time duration, $\Delta x, \Delta t$ are spatial and temporal resolution, subscripts *raw, out* indicate raw data and filtered outputs, respectively, σ, τ, a are the spatial parameter, the temporal parameter and the width factor of the kernels used in the ASM filter (Schreiter et al., 2010a).

Algorithm	Complexity
Conventional	$\mathcal{O}(X \times T \times \frac{\sigma \tau a^2}{\Delta x_{out} \Delta t_{out} \Delta x_{raw} \Delta t_{raw}})$
Cross-correlation	$\mathcal{O}(X \times T \times \frac{\sigma \tau a^2}{\Delta x_{out}^2 \Delta t_{out}^2})$
Fast Fourier Transform	$\mathcal{O}(X \times T \times \frac{1}{\Delta x_{out} \Delta t_{out}} \times \log \frac{XT}{\Delta x_{out} \Delta t_{out}})$

Congestion marking The complexity of the proposed detection method for link congestion can be divided into different steps as follows.

1. Congestion/Free-flow masking: comprises of comparison operations (with the related thresholds, respectively) at all nodes. Hence, the complexity of this step is $\mathcal{O}(|X| \times |T|)$.
2. Congestion growing: can be done by a convolutional operation of the congestion mask and the chosen (growing) kernel. Assuming the size of the kernel is α , the complexity of this step is $\mathcal{O}(\alpha \times |X| \times |T|)$
3. Strimming: is simply an element-wise masking operation, hence, the complexity is equivalent to the number of elements, meaning $\mathcal{O}(|X| \times |T|)$.

It can be seen that all the operations are done in the complexity that is equivalent to the size of the processed data. In summary, the complexity of the link congestion detection is $\mathcal{O}(|X| \times |T|)$.

Congestion graph simplification All nodes and links are only needed to processed once. That leads to the complexity of this module being $\mathcal{O}(|E^c| \times |V^c|)$.

Congestion clustering A depth-first search algorithm is used, hence, the complexity of this component is linear to the size of the congestion graph, i.e. $\mathcal{O}(|E^c| + |V^c|)$.

Congestion-path tracking Algorithm 3.1 aims to identify all possible expansions from a given node, both spatial and temporal dimensions are taken into consideration. By introducing the variable *isVisited*, each node is traversed only once. Also, each link is checked once as well. Hence, the complexity for the traversing is proportional to $|E^c| + |V^c|$. Another complexity of the algorithm is operations after paths and sub-graphs have been learned from upstream nodes (successors). They are combined into the final (path/sub-graphs) sets of the current node. As this is similar to what happens in the next component, the complexity of this part is discussed in detail in the following.

Overlapping-path merging Algorithm 3.2 combines paths and sub-graphs into a final collection. In principle, each path/sub-graph is compared with all other paths. This leads to a total of N_p^2 comparisons, in which, N_p is the number of paths found by all origin nodes, i.e. all paths in M after traversing the weakly connected components of the congestion graph. It is not trivial to analyse the complexity of this component since N_p is strongly dependent on both the characteristics of the graph and how related congestion evolves. A worst case could lead to exponential complexity. However, the problem can be compromised by utilizing a great deal of memory to index all possible paths in the spatially projecting graph. Nevertheless, to avoid unnecessary complications, computation time could be evaluated to check whether improvements are required. This is reflected in the case study described later.

Conclusion The overall complexity is determined mainly by that of the ASM method and the number of comparisons in merging overlapping paths. The former is linear to the size of the filtered data, while the latter might be exponential w.r.t. the graph size of the related cluster of congestion. The complexity of the latter term depends solely on the realised congestion. Our experiment suggests small numbers of congestion paths in M , hence, we can expect a dominant significance of the former in general. Accordingly, it can be expected that the proposed methods are scalable with the size of the underlying (highway) network due to the linear complexities of all the proposed components (except for the overlapping-path merging).

3.4 Case study

This section presents a case study, in which the proposed framework is applied to collecting congestion patterns from a nationwide traffic network. We analysed the performance of the framework in the following aspects.

- Computation: the time efficiency of the developed framework.
- Correctness: the ability to form congestion patterns that satisfy the condition C2 in Definition 3.1.
- Quality: some insights into the quality of collected patterns.

3.4.1 The Netherlands highway network

In this study, the highway traffic network in the Netherlands is used to evaluate the developed framework. This network is one of the densest highway networks in the world. It is approximately 64 km per 1000 km² and includes more than 2500 km of roadways (Netherlands Transport, 2020).

The geographical representations (shapefiles) of the Dutch highway network are publicly accessible (Ministry of Infrastructure and Water Management, 2020). Given one of these files, an equivalent graph representing the network topology is constructed, which serves as one of the main inputs of the framework. Specifically, the shapefile in June 2018 is arbitrarily chosen for this study. The resulted graph after preprocessing has 9592 nodes and 11595 edges.

3.4.2 Traffic data

In this study, traffic data from loop detectors are used. They are implemented at around 10000 locations on the Dutch highways. Fig. 3.4 gives a general idea of the distribution of these detectors on the Dutch highway network. Vehicular speeds and flows are logged every minute which constitutes the raw dataset. The data on June 01, 2018, are retrieved for extracting congestion patterns.

The raw data are filtered by the ASM method as described in our framework. In particular, 200 meters and 30 seconds are chosen as spatial and temporal resolutions, respectively, of the estimated data.

3.4.3 Computational resource

Regarding the power resource, we ran our experiment on a desktop computer equipped with an Intel(R) Xeon(R) CPU E5-1620 v3 @3.50 GHz. The CPU has 4 physical cores

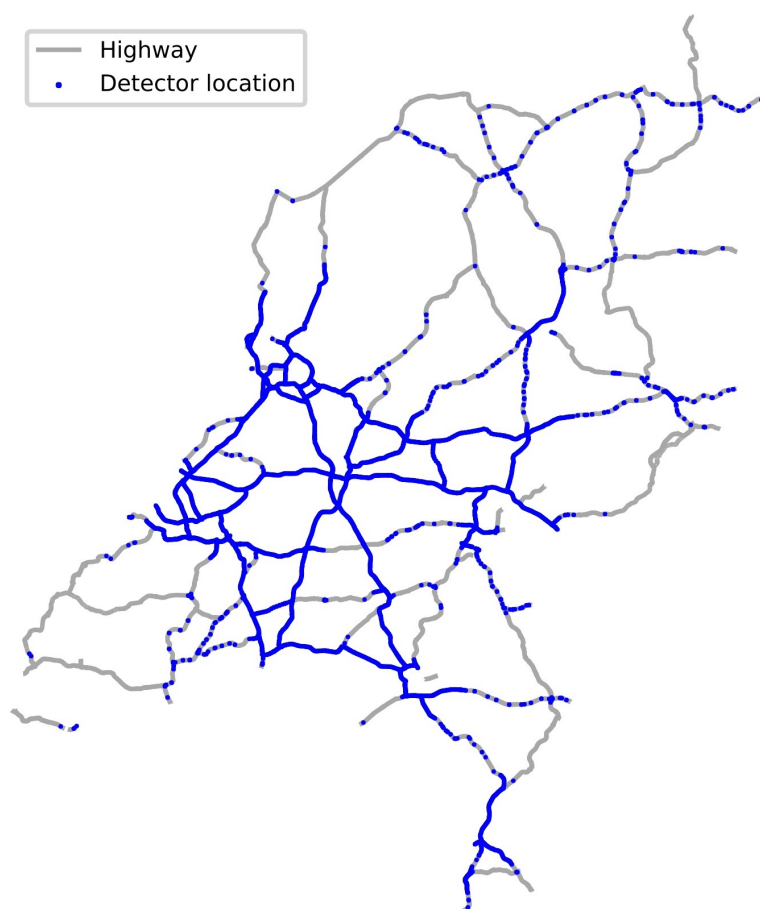


Figure 3.4: Distribution of loop detectors on the Dutch highway network

which are divided into 8 logical processors. The random access memory (RAM) is of 16 GB capacity.

3.4.4 Results

Practical efficiency

Various processing times are provided in Table 3.2. The whole process took around 35 minutes to identify and construct all congestion patterns. The most expensive task, data retrieval, is related to input/output (i/o) access, which is highly dependent on how measurements are stored. In specific, a great deal of time was spent on map-matching detector locations to retrieving paths. The network partitioning was done in 4 seconds (note that this step only needs to be executed once). It took 178 seconds for the ASM method to filter the raw data. Congestion marking took 231 seconds. The congestion graph was processed in less than 1 minute in total. For a nationwide traffic network, it can be concluded that this performance is practically efficient.

The majority of the execution time was spent on those components whose input data are as large as filtered data (i.e. X, T in total lengths and time intervals). The tasks related

Table 3.2: Executing time of various proposed components for collecting congestion patterns during the June 1, 2018

Component	Executing time (seconds)
Network partitioning	4
Data retrieval	1489
Traffic speed reconstruction	178
Congestion marking	231
Graph simplification	11
Network traversal	15
Overlapping-paths merging	14
Total	1942

to the congestion graph was done much more quickly, in particular, 41 seconds (as compared to 409 seconds). This suggests that improving Algorithm 3.1 or Algorithm 3.2 is not a priority. In addition, this further strengthens the scalability of the proposed framework and related methodologies to larger-size traffic networks.

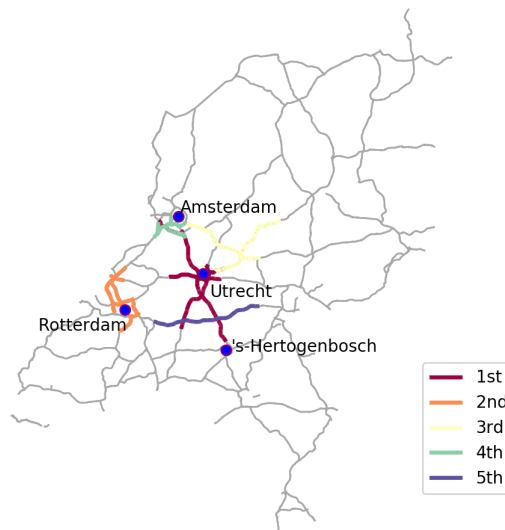


Figure 3.5: The five largest clusters of congestion identified on June 1st, 2018

Intermediate outputs

The network partitioning step results in 1531 paths, of which, 593 paths have loop detectors implemented. From these paths, raw data are retrieved accordingly. The congestion marking (which is applied on the ASM-filtered data) results in a congested graph with 611523 nodes and 2002344 links. This graph is significantly simplified to a graph with 26355 nodes and 169346 links, which are equivalent to reductions of 95% and 91%, respectively.

For the studied day, 472 clusters of congestion are formed by identifying connected components of the congestion graph G^c . Fig. 3.5 highlights the five largest clusters found in terms of total extent distance. They are mostly located in the Randstad area which includes many major cities, e.g. Amsterdam, Rotterdam and Utrecht. The most critical cluster covers nearly 240 kilometres of roadway and occurred during the morning hours and covered a large part of the A2 highway, which is one of the busiest roads in the Netherlands. The cluster expands to three major cities, namely Amsterdam, Utrecht and 's-Hertogenbosch.

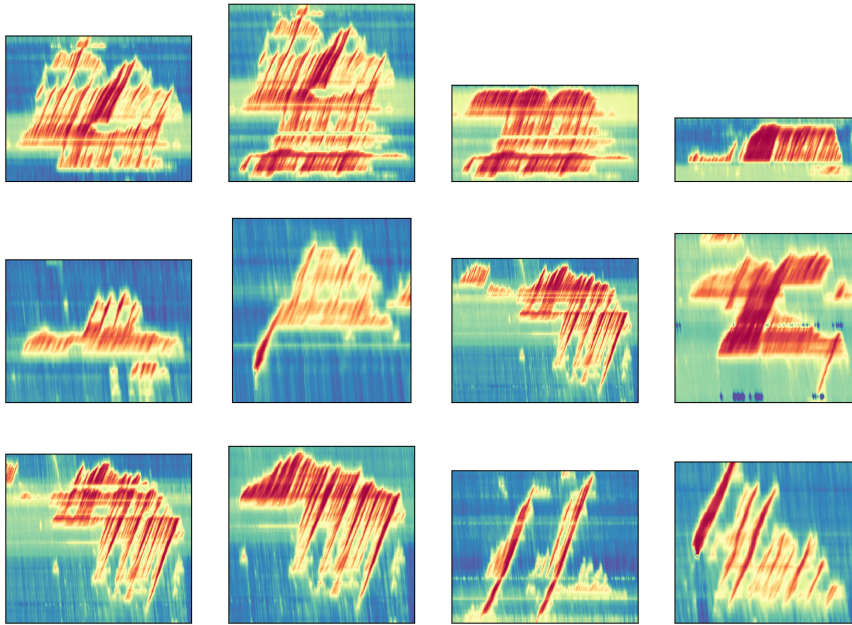


Figure 3.6: A few examples of congestion patterns collected from data on June 1st, 2018

From the identified clusters, 949 spatial-temporal patterns of congestion are constructed. Fig. 3.6 shows a few examples from the pattern collection. Different appearances of congested traffic are involved, such as singularly wide moving, stop-and-go, homogeneous and complicated congestion. With these patterns ready at hand, the resulting dynamics of congestion at different locations on a traffic network are more intuitively observed. In addition, possible relations of traffic between different locations are visualized in these 2D patterns.

Fig. 3.7 (a) shows one of the complex patterns found, which is approximately 55 kilometres in travelled distance. The topological plot shows that the pattern is constructed by successfully stitching congested regions from different paths (which are resulted from partitioning the network). The congestion involved several bottlenecks. Two of them were strongly congested. The downstream bottleneck caused a number of disturbances which only resolved after reaching the farthest upstream bottleneck. (see Fig. 3.7 (b)).

As shown in Fig. 3.7, there is a region of congestion at the upper-left corner. To a

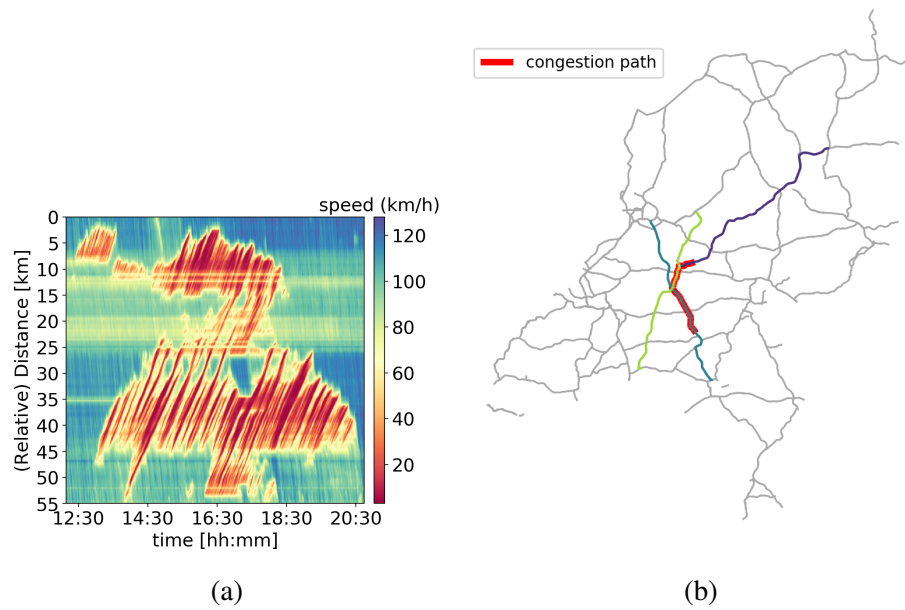


Figure 3.7: An example of congestion pattern on June 1st, 2018, (a) 30-second-aggregated vehicular speeds and (b) topological information of the related path as well as the underlying original paths, (of which links are stitched to form this congestion path), shown in colours other than red

certain extent, this can stand as an independent pattern since it is not connected with the remaining congestion regions in the pattern. The reason for obtaining this entire image as one pattern is simply the employed congestion detection. More sophisticated methods can be applied to obtain more finely isolated patterns. The trade-off is mostly the required computation time.

Overlapping patterns

Condition C2, pattern completeness, in Definition 3.1 assures a complete view of congestion is provided by any pattern. Together with topological information, this completeness is necessary for analysing the congestion. However, this can result in patterns that are highly overlapping. Fig. 3.8 gives an example extracted from the resulted patterns. Approximately 90% of the two patterns are the same. An investigation into the topologies of those two paths shows that the difference is due to the existence of parallel roads. It should be noticed that whether to avoid overlapping patterns is a design choice and probably depends on target applications. Nevertheless, if deciding against overlapping, one can introduce various criteria on the final congestion paths, such as paths need to have a certain percentage of distinction, for instance, 80 per cent.

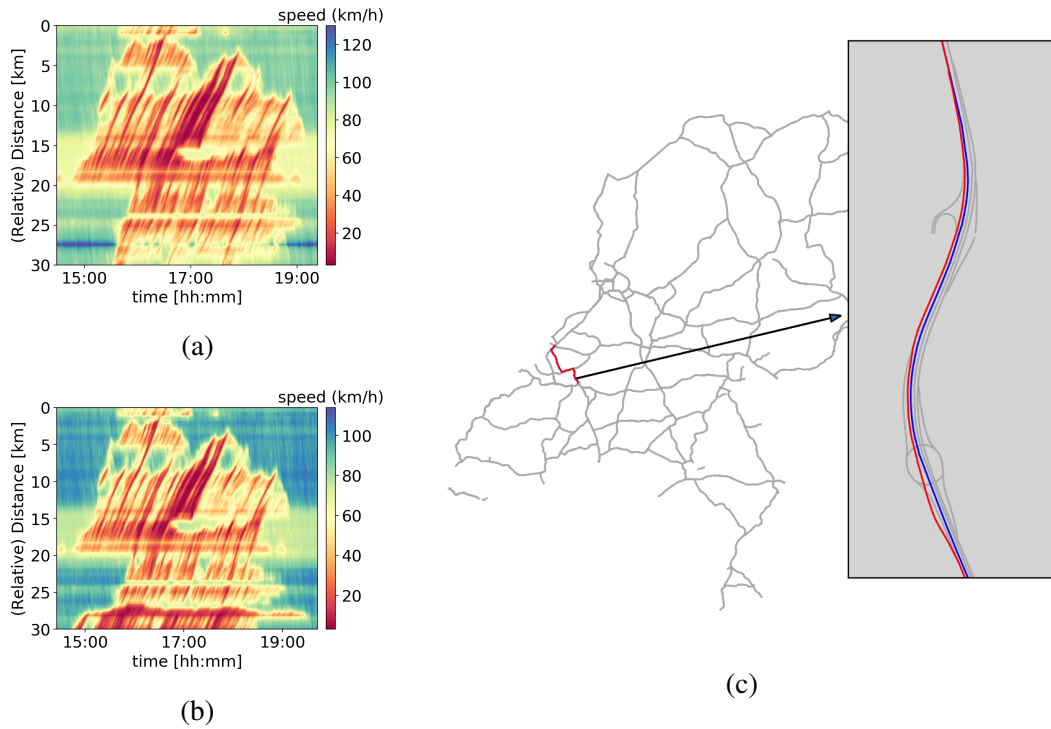


Figure 3.8: An example of overlapping patterns. (a,b) speed maps, (b) the topological information of the two underlying paths. The zoomed-in section (in grey colour) indicates the starting part of the difference between two paths. Specifically, one path contains the main carriageway along the eastern ring of Rotterdam and the other path includes part of the parallel carriageway.

3.5 Conclusion

This chapter has presented a framework and related methods to mine traffic data for patterns of congestion. Each pattern represents the evolution of traffic congestion both spatially, i.e. how it propagates upstream, and temporally in a complete fashion, i.e. from the onset until the recovery of free-flow traffic. The task is achieved by processing through three phases, namely congestion detection, congestion clustering and patterns extraction. In the first phase, by cutting a traffic network into individual paths, traffic speeds over the network are effectively reconstructed which results in a fine resolution of (vehicular speed) data. This outcome is beneficial to the detection of congestion on each path. The second phase utilizes the graph concept and the underlying topology of the traffic network to associate congestion from different paths. This leads to a three-dimensional graph where each node represents congestion at a specific place and a particular time. A connected component of this graph represents a cluster of congestion, e.g. traffic jams from multiple locations reaches and mixes at upstream locations. These components are identified and isolated for the tracking of corresponding congestion paths in the third phase. A graph-based traversing algorithm is developed to follow congestion while pertaining to only one single path.

The analyses of various components indicate an expectation of linear complexity of the overall framework. In particular, it is scalable with the size of the underlying highway network. The case study on the Dutch national highways suggests that the framework is applicable for large-scale traffic network. The method has found a number of different clusters of congestion, from which many congestion patterns are tracked and extracted successfully.

In the next chapters, characteristics of congestion patterns are studied using different approaches. The objective is identify salient features that can match/contrast similar/different patterns.

Chapter 4

Feature Extraction (i) - Generic Key-points versus Traffic-related Patterns

Determining salient characteristics of traffic congestion (in spatiotemporal representation for specific) plays a vital role in discriminating different patterns. The extracted features are beneficial to the development of machine learning methods to classify and/or label congestion patterns automatically. In this chapter, we analyse two fundamentally different feature approaches, namely generic and domain-specific. While the former inherits the knowledge of general image classification, the latter looks for characteristics from a traffic-domain point of view.

This chapter is an edited version of the following published paper:

Nguyen, T. T., P. Krishnakumari, S. C. Calvert, H. L. Vu, H. Van Lint (2019) Feature extraction and clustering analysis of highway congestion, Transportation Research Part C: Emerging Technologies, 100, pp. 238–258.

4.1 Introduction

Highway traffic congestion is one of the central aspects concerning mobility management. To study it, traffic data has been collected for decades using various sensory systems such as inductive loops, AVI - Automatic Vehicle Identification and FCD - Floating Car Data. This includes relevant important traffic indicators like speed, flow or travel time which can be used for various purposes by road administrators, industry and academia, including policy evaluation (e.g. Van Lint et al., 2008; Zheng et al., 2012), traffic management (e.g. Soriguera & Robusté, 2011; Vlahogianni et al., 2005), traffic modelling and simulation (e.g. Wang et al., 2006; Spiliopoulou et al., 2014), to name just a few applications. National Data Warehouse (NDW) ndw, for instance, is a Dutch organisation that has been doing data collection for almost ten years over more than 7000 kilometres of freeways and provincial roads in the Netherlands. They have now stored more than 200 TB of traffic data which is a huge informative source to understand traffic dynamics.

Massive amounts of traffic data are a rich source of information; however, they also pose a challenge on how to manage them efficiently, for example regarding fast access and search-ability. Classification, i.e. adding meaningful labels to the data, is an essential step to enhance the utilisation of the storage and also to gain insights in the types of traffic congestion patterns that can be found in the collected data. A well-indexed and labelled dataset, for instance, can result in efficient search engines, which are essential for data retrieval. Both processing time and accuracy are of significant concern in this regard. The simplest way is to annotate traffic patterns manually; however, manual classification is likely only suitable for datasets with limited amounts of items since manual annotation is time-consuming and susceptible to bias. For large size datasets, an automatic approach is undoubtedly required.

Classification of highway traffic congestion has been conducted in different ways (as shown in the literature), mainly focused on either a theory-laden or data-driven approach. Theoretical approaches explain congestion by mathematical equations which describe relationships between fundamental traffic variables such as speed and flow. Hence, different taxonomies can be derived in association with supporting theories. Schönhof & Helbing (2007) simulate traffic using the nonlocal gas-kinetic based model to produce five different types of congestions - pinned localized cluster, moving localized cluster, Stop-and-go waves, oscillating congested traffic and homogeneous congested traffic. These patterns are also found in their empirical studies. Kerner (2002) defines a different taxonomy based on his three phase traffic theory. It constitutes of two fundamental types - general pattern (GP) and synchronized pattern (SP). Although these classifications are underpinned by solid conceptual and mathematical ideas about traffic dynamics, no automated methods for applying them on traffic patterns have been reported yet. In contrast, data-driven approaches instead focus on similarities between congested patterns represented in traffic data. Machine learning approaches in this direction provide more opportunities for automation. A recent idea is to conceive

spatio-temporal maps as images (Nguyen et al., 2016; Krishnakumari et al., 2017). By doing so, advanced computer vision methods for image classification can be employed to classify traffic patterns. In Nguyen et al. (2016); Krishnakumari et al. (2017), a supervised learning approach was adopted, i.e. classifiers were trained based on manually labelled datasets. These works also show that lacking of traffic-domain knowledge (Nguyen et al., 2016) or simple image processing like naive contour extraction (Krishnakumari et al., 2017), can lead to low accurate levels of classification.

Clustering analysis is unsupervised learning which aims to find an intrinsic partition of a dataset without any labelled items. Using spatio-temporal representations of highway traffic data like speed or flow, different elements of congested traffic can be noticeably observed. Thus, we aim to employ clustering analysis on highway congestion. By using images to represent traffic data, visual features can be extracted. To this end, this chapter proposes and compares two inherently different feature extraction methods, namely point-based and area-based approach. While the former searches for automatic features which are motivated by computer vision, the later extracts domain knowledge-driven characteristics based on image segmentation. The case study demonstrates the ability of Watershed technique (Beucher & Meyer, 1992) in segmenting image (congestion) patterns into different segments on which corresponding features can be extracted sufficiently. In addition, by conducting cluster analysis on a dataset, a hierarchical representation of congestion with respect to their similarities is constructed. From which, typical patterns in the dataset are explored, and an initial categorization of the dataset can be created automatically and efficiently. Accordingly, appropriate labels can be generated for annotating congested patterns automatically. Finally, the effectiveness of domain knowledge in pattern representation is shown by comparing the two feature schemes.

The rest of the chapter is organized as follows: Section 2 presents the literature review on the classification of congestion patterns. Section 3 represents the methodology in detail, which comprises of two approaches. Next, available data and evaluation metrics for clustering analysis are described in Section 4. Then, results and discussion are provided in Section 5. The conclusion in Section 6 justifies the work and proposes further research.

4.2 Literature Review

There have been a significant number of studies investigating traffic dynamics with a focus on describing and understanding the resulting spatio-temporal traffic patterns. One of the main objectives is to discern different states of congested traffic. In general, there are two approaches for this, namely theory-laden and data-driven. The theoretical approach represents traffic dynamics mathematically. Related models are validated by their abilities to reproduce congested states or patterns that are observed in real life. On the other hand, the data-driven approach explores the various observed congestion

patterns and analyses their characteristics further. This section reviews some of the related works and provides discussion on the ability to automate classification. Fig. 4.1 provides a taxonomy of research on this topic in the literature.

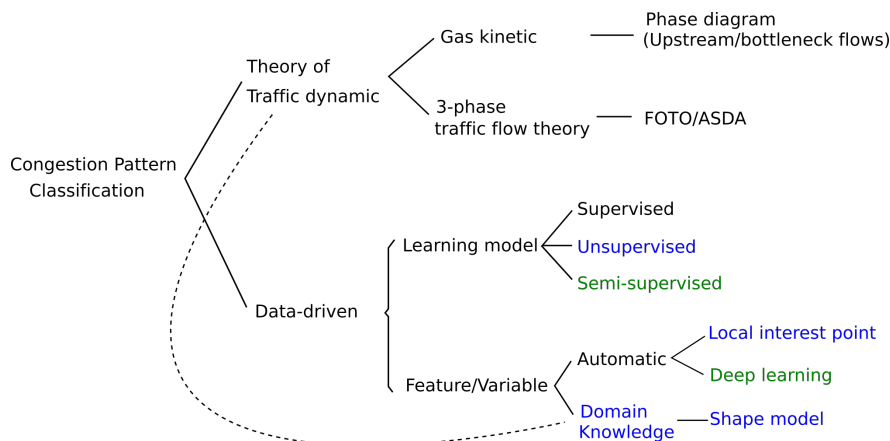


Figure 4.1: Taxonomy of research on classification of congestion patterns. Color scheme code: black-existing research, green-possibility and blue-elements used in this study.

A notable study in the theory-laden approach is performed by Schönhof & Helbing (2007), which employs a gas-kinetic model, which is a second-order model, to simulate traffic flow. By manipulating the relation between upstream flow (Q_{up}) and bottleneck strength, e.g. on-ramp flow (ΔQ), the model is able to reproduce five typical congested patterns, which are also confirmed by empirical data - namely pinned localized cluster (LC), moving localized cluster (MLC), Stop-and-go waves (SGW), oscillating congested traffic (OCT) and homogeneous congested traffic (HCT). Subsequently, a 2D phase diagram is constructed which correlates different congestion patterns with combinations of Q_{up} and ΔQ . This concept of analyzing these two variables is also investigated in previous research (Helbing et al., 1999; Lee et al., 2000). Although there is a solid conceptual and mathematical foundation for the method, there are two challenges for applying this phase diagram to real traffic data. Firstly, estimating (Q_{up}) and (ΔQ) is not a trivial task, and there is no accurate methodology that has been validated yet. Secondly, assuming the information on types of congestion is given, as in (Schönhof & Helbing, 2007), the obtained empirical phase diagram shows high interference between different congested traffic patterns. Therefore, the application of this phase diagram in the classification of traffic patterns (with real data) is not feasible.

Another well-known study is done by Kerner (2002) which is based on his proposed three phase traffic flow theory. Two main categories of congested traffic at freeway bottlenecks were introduced, namely general pattern (GP) and synchronized pattern (SP). Variations of congestion are asserted to emerge from these two main classes. According to this theory, traffic exists in one of three states, namely "free flow", "synchronized" and "wide moving jam". For recognizing and monitoring congestion patterns, Kerner developed two models - FOTO (Forecasting of Traffic Objects) and ASDA

(Automatic Tracking of Moving Traffic Jams). These two models identify and keep track of traffic in "wide moving jam flow" and "synchronized flow" respectively. Consequently, every traffic state in spatio-temporal maps is automatically classified into one of the three states. There are two critical points which might limit the expansion of these models into traffic congestion classification. First, one of the underlying foundations of these models is the set of various fuzzy rules, (4 and 13 for the basic and extended sets, respectively), which consist of a number of parameters. Their values are found solely based on experiments (Kerner et al., 2004). Hence, to apply them to a different road or highway, they need to be calibrated by properly understanding the model and conducting experiments with empirical data. A systematic methodology for defining and calibrating the model still needs to be developed. Second, these models only focus on classifying traffic into three states, from which a qualitative analysis can be conducted to distinguish different spatio-temporal congested traffic, for instance, the GP or SP pattern. As for automating the classification of congestion patterns, no automatic method based on these three phases has been reported yet.

The data-driven approach constitutes two vital components - learning model and feature extraction. Depending on the availability of data, i.e. ground truth or labelled patterns, there are three learning models: supervised, unsupervised and semi-supervised (Witten et al., 2016). The supervised learning infers a pattern-label mapping function by learning from examples in a training dataset, while unsupervised learning does not require labelled patterns and explore connectivities between patterns instead. The semi-supervised learning aims to leverage the unlabelled data to improve the performance of the learners which are inferred from a few labelled items. The second component, feature extraction, concerns how to represent a pattern using specific variables. They are ideally chosen in a way to increase the chance of discriminating different patterns. Very few studies follow this approach and most (if not all) of them used supervised learning method. In (Nguyen et al., 2016), a dataset of congested patterns is manually partitioned into five different classes namely isolated wide moving jam, large scale heterogeneity I, large scale heterogeneity II, homogeneous and mixed class on traffic data from two of the busiest highways in the Netherlands. This study applies automatic feature extraction in computer vision to traffic patterns. Then, a classifier based on support vector machine (Cortes & Vapnik, 1995) is trained on this dataset to classify traffic patterns. Another approach describes contours of congested patterns by shape model (Krishnakumari et al., 2017). The authors to some extent take traffic knowledge into account, i.e. using wide moving jam as one of the features. The classification accuracy is not high due to the naive contour extraction.

One of the obstacles in approaching with supervised learning is the availability of a training dataset, specifically labelled patterns. Manually processing numerous traffic congested objects is time-consuming and non-efficient as human judgment is ordinarily subjective and inaccurate. This can be tackled by applying methods from unsupervised or semi-supervised learning approach. In particular, clustering analysis is a well-studied subject in the field of unsupervised learning which aims to divide a given

dataset into different groups of items, so-called clusters. Each cluster can represent a typical pattern of the dataset. Despite the absence of a precise definition, the underlying principle is to maximize the similarity between items in the same cluster and to also maximize the dissimilarity of items from different clusters. Its applications encompass various domains and fields - data mining, text analysis, information retrieval, data annotation and pattern recognition to name a few (Xu & Wunsch, 2005). An extensive body of literature on existing clustering algorithms can be found in (Xu & Tian, 2015; Fahad et al., 2014; Jain, 2010).

In the traffic domain, clustering analysis has been applied broadly to explore different datasets. For example, Depaire et al. (2008) cluster traffic accidents into different types given related variables such as vehicle type, gender and road type. The authors found that the employed clustering model is beneficial to the followed-up injury analysis by, for example, revealing new influenced variables for the injury outcome. Kim and Mahmassani extend a density-based clustering algorithm to cluster vehicle trajectories (Kim & Mahmassani, 2015a,b). Representative trajectories are then determined for the obtained clusters which show major traffic flow patterns over a network. Celikoglu & Silgu (2016) uses multivariate clustering, (as an extension to their previous dynamic classification approach (Celikoglu, 2013)), to partition flow patterns over the fundamental diagram. The method is able to capture sudden changes or transitions of flow patterns between successive times which are promising for non-recurrent congestion detection and control. These studies have shown advantages of clustering analysis in exploring potential partitions of data, where applicable. Obtaining clusters can also assist experts in doing classification and interpreting datasets at a higher level.

This chapter aims to explore the potential of unsupervised learning in traffic congestion classification. The outcomes are expected to bring first insights into a database of patterns and help annotate those automatically which certainly saves a lot of manual labelling effort. We further develop and analyse the two approaches to feature extraction - automatic and domain knowledge related.

4.3 Methodology

4.3.1 Overall framework

Given a set of raw traffic speed data, the goal is to obtain an automatic classification of congested patterns. For this reason, a scheme for representing traffic patterns needs to be developed in which certain characteristics are extracted. In this chapter, two different approaches are described and compared. They both conceive patterns as general images and apply different computer-vision methods to process them. One approach searches for generally local features while the other explores traffic-related characteristics. A clustering method then can be applied based on these features in an attempt to

find reasonable structure of the given dataset. Fig. 4.2 illustrates the proposed framework including three main steps: (i) preprocessing data, (ii) extracting representative features from a pattern, and (iii) clustering the dataset based on these features. For the sake of clarity, this section briefly describes various parts of this framework.

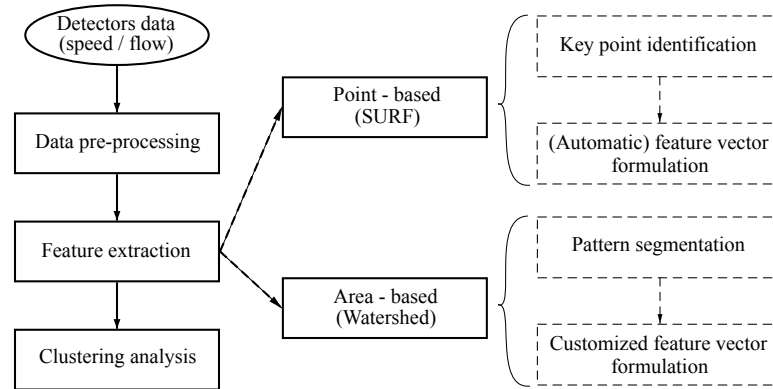


Figure 4.2: General framework for pattern clustering

Data preprocessing

Spatio-temporal maps have been used extensively to gain insights into traffic dynamics. Particularly, information such as speed or flow of a road stretch during a certain time period can be analyzed at a broader perspective, i.e. complete space-time view, than at just local detectors. An example of such spatio-temporal map is shown in Fig. 4.3a. The speed measurements are collected from the detectors distributed sparsely along the road stretch over fixed intervals.

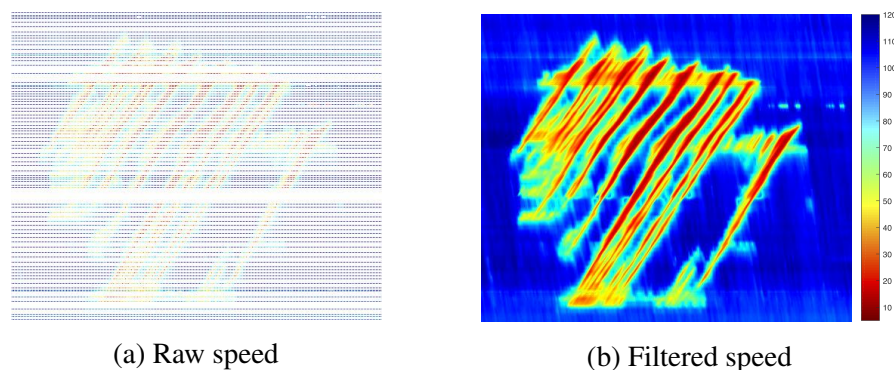


Figure 4.3: Spatio-temporal speed maps of traffic on A12 highway in the Netherlands on April 12th, 2016, from 06:30am to 10:30am. The horizontal and vertical axes represent time and detector locations respectively. The driving direction is from top to bottom. For visualisation purpose, a colormap is used to emphasise different patterns inside this congestion, e.g., wide moving jams. Colors code: red implies low speeds (congested condition) while blue is for high speeds (free flow condition).

As seen from Fig. 4.3a, instant speeds are represented by a corresponding point in the figure. This poses a question about traffic dynamic around those detectors where no data is available. The well-known adaptive smoothing method (ASM) serves the purpose of interpolating the unseen traffic data. For details about this technique, we refer the reader to (Treiber & Helbing, 2002; Van Lint & Hoogendoorn, 2010; Schreiter et al., 2010a). The result of applying this filtering technique to the raw speed is shown in Fig. 4.3b. The advantage of ASM is two-fold. Firstly, ASM fills in missing values and smooths out (high frequency) noise in raw speed measurements. Secondly, detectors are implemented at locations which are not necessarily at equal distances (see gaps between horizontal lines in Fig. 4.3a). The ASM produces an equidistant grid of smooth speeds (see Fig. 4.3b). Both of these points support better application of image processing techniques in the feature extraction step.

For consistency throughout the chapter, we define a *pattern* of congestion as a numeric array of speeds representing (congested) traffic states over spatial and temporal dimensions. These patterns can be visualised using heat maps as shown in Fig. 4.3 (left: based on raw data; right: result of ASM filtered speeds). Since throughout the chapter we use ASM filtered speeds, a congestion pattern in the ensuing is based on such filtered data.

Feature extraction

Feature extraction is one of the crucial steps to obtain an efficient representation of input patterns for data mining applications like clustering or classification. The key is to identify distinct features that make traffic patterns distinguishable from each other. The high quality of ASM filtering result (Fig. 4.3b) motivates a vision-based approach in which spatio-temporal patterns of congested traffic are conceived as grayscale images (or intensity images). Each pixel intensity is assigned by the corresponding speed value. These values are scaled to the range $[0; 255]$ to fit with 8-bit representation of grayscale images. Notice that, they are, in essence, numeric matrices of filtered speed measurements. By considering them as grayscale images, we can apply advances of computer vision to traffic field.

We consider two approaches from different domains and compare their performances in clustering traffic patterns - namely point-based features and area-based features as shown in Fig. 4.2. As its name suggests, the point-based method explores local features which are in images. Some examples include corners, blobs or locations where pixel intensities change sharply. On the other hand, the area-based method is more about higher level features such as shapes or areas to build a customized feature vector which could potentially later incorporates traffic domain knowledge. Further details of these two methods are given in Section 4.3.2 and 4.3.3.

Clustering analysis

The clustering aims at finding typical congestion patterns in a dataset. Overall, there are two approaches for clustering analysis, namely hierarchical and partitioning (Jain, 2010). We have chosen the hierarchical clustering as our main clustering method for two reasons. Firstly, this approach constructs a hierarchical representation of a given dataset which, in turn, provides an overview of the distribution of existing congestion patterns. Secondly, hierarchical clustering provides the ability of reproducibility of resulting clusters. This avoids the sensitivity to random initiations that most of partitioning clustering methods, e.g. k-means, encounter.

Hierarchical clustering can work in two different fashions - agglomerative and divisive - which are inherently bottom-up and top-down strategies for constructing a binary tree. The latter considers the entire dataset as one cluster and gradually divides it into smaller clusters. Since there are $2^n - 2$ ways of splitting a cluster of n patterns, heuristic rules are normally applied. In contrast, the former initiates each pattern as a single cluster and examines connectivities between patterns or intermediate clusters. The underlying idea is to combine the two closest patterns or (intermediate) clusters into a new cluster. Hence, we employ an agglomerative approach to conduct clustering analysis. The connectivity between any two patterns is calculated using Manhattan distance, also known as *city block*. For the distance between two clusters, the average-link scheme is implemented, which takes the average distance between all pattern pairs from those clusters (see Eq. 4.1). This process continues until only one cluster remains, meaning all patterns are in the same cluster.

$$d(p, q) = \sum_{l=1}^D |p(l) - q(l)|$$

$$d(\mathcal{R}, \mathcal{S}) = \frac{\sum_i^{|\mathcal{R}|} \sum_j^{|\mathcal{S}|} d(x_i^{\mathcal{R}}, x_j^{\mathcal{S}})}{|\mathcal{R}| \times |\mathcal{S}|} \quad (4.1)$$

Where,

p, q are two feature vectors representing two patterns which have D dimensions

$p(l)$ is the l _th element of vector p

\mathcal{R}, \mathcal{S} are two clusters

$|\mathcal{R}|, |\mathcal{S}|$ are the numbers of patterns in cluster \mathcal{R}, \mathcal{S} respectively

$x_i^{\mathcal{R}}$ is the feature vector representing the i _th pattern in cluster \mathcal{R}

4.3.2 Point-based feature extraction

Using point-based features is one of the most common practices for extracting image characteristics in computer vision. The point-based features are known as interest

points (or key-points) – which indicates distinctive locations in an image that contain prominent local information. As mentioned previously, the work in (Nguyen et al., 2016) demonstrated the ability of this method on classifying different traffic patterns. Here we summarize this method and also refer to the original work for further details.

Generally, the point-based method constitutes two main steps, namely (i) key point identification and (ii) feature vector formulation. It is indicated so in Fig. 4.2. The first step explores important points in an image, i.e. traffic pattern, which are defined by their local features. A collection of these points is expected to distinguish different images. Given detected points, the second step formulates a feature vector for pattern representation.

Key point identification

The key point identification step comprises of two pivotal elements - key point *detector* and key point *descriptor*. The key point detector is responsible for identifying interest points in an image. They present at various special locations such as L-corners, blobs, T-junctions or Y-junctions. These points have been widely used as local features for distinguishing between images. Several examples of such points are indicated by small circles on three different traffic patterns shown in Fig. 4.4. It can be easily observed that these key points usually occur around the edges of these traffic patterns. A number of different detectors can be found in the literature (Harris & Stephens, 1988; Lowe, 1999; Kadir & Brady, 2001). Amongst those, the Fast-Hessian detector is claimed to be highly accurate and fast (Bay et al., 2008). The underlying idea is to find strong changes in surface curvature represented by intensities in an image. For that, the so-called *Hessian matrices* are constructed using second-order derivatives of the image (see Eq. 4.2). $I(x,t)$ represents speed values over time and space dimensions. The determinant of the Hessian matrix is used as the indicator of key points in the input image. In addition, such determinants are also calculated in different scales of input images (so-called *scale space*) to assure the repeatability of detecting same key points in images at different sizes. A non-maximum suppression is then applied to identify maxima of determinants in 3D space (space-time-scale) as key points of the image.

$$H(x,t) = \begin{bmatrix} \partial^2 I(x,t)/\partial x^2 & \partial^2 I(x,t)/\partial x\partial t \\ \partial^2 I(x,t)/\partial x\partial t & \partial^2 I(x,t)/\partial t^2 \end{bmatrix} \quad (4.2)$$

The second element, key point descriptor, aims to construct a distinctive representation for key points found previously. In generally, this is done by applying an appropriate method to represent surrounding of corresponding points. Speed-Up Robust Feature - SURF - descriptor (Bay et al., 2008) is adopted in this chapter. Given a key point at (x,t,s) , a window with the size of $20s$, (s indicates the scale at which the corresponding key point is found), is formed surrounding the key point. Then, the neighbourhood determined by the window is divided into 4×4 sub-regions on which Haar wavelet transform (Chui, 2016) responses are performed. This results in a 64-dimensional

real valued feature vector for any given key point (the so-called SURF-64 (Bay et al., 2008)). We refer the readers to (Bay et al., 2008) for a comprehensive description and parameter analysis related to this method.

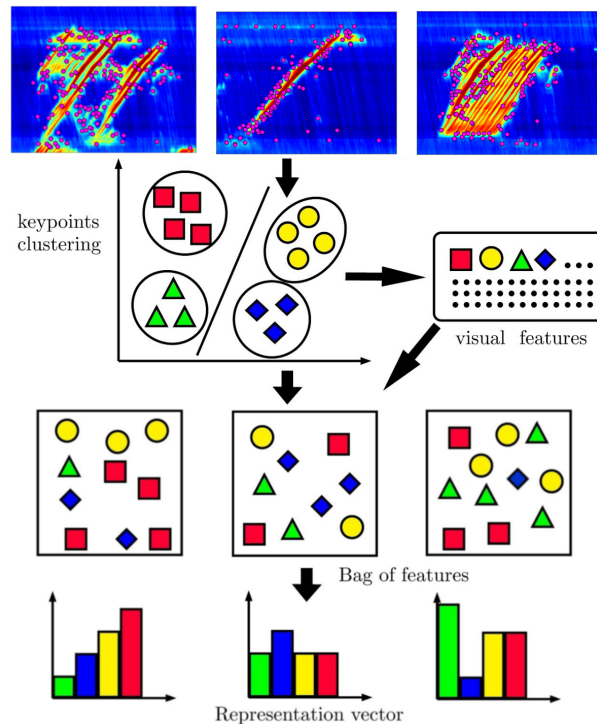


Figure 4.4: Point-based feature extraction using the bag-of-visual-word method. This figure is reproduced from Fig. 5 in (Nguyen et al., 2016).

Feature vector formulation

The first step results in a number of interest points in an image pattern. They are represented by a 64-dimensional feature vector. The feature vector formulation step describes a method to build representations for image patterns by accommodating these key points. It adopts the "bag-of-word" idea which is a well-known method in the topic of document classification where a document is represented by a set of its words. In that way, word order or any combinations are discarded and only their occurrence frequencies are preserved. When it comes to computer vision, it is also known as "bag-of-visual-word".

The "bag-of-visual-word" comprises of two main steps. It starts by constructing a *visual word dictionary* - an analogy with word dictionary. Since the feature vector representing key points consists of 64 real values which are continuous, they are grouped into discretised groups to facilitate the construction of such dictionary. This is illustrated in the second row of Fig. 4.4. In this framework, k-mean clustering method is implemented to partition all the key points collected from all image patterns. The number of clusters is specified a priori. By doing this, similar points are expected to

be in the same groups and they all only represent a typical pattern of key points. Consequently, a visual word dictionary is constructed which has size as number of groups and each key in the dictionary is a typical key point, so-called *visual word*. For example, in Fig. 4.4, points are divided into 4 groups with different shapes/colors; hence, the corresponding dictionary comprises of 4 different visual words.

Next, all key points found in an image pattern are matched with visual words in the constructed dictionary. This is a classification problem in which we need to classify each of the detected key points to one of the visual words. A k-nearest neighbour (Cover & Hart, 1967) (knn) based classifier is trained on the clustering results from the previous step, which classifies a point into one of the visual words. Notice that, for those patterns which are used to construct the visual word dictionary, the clustering step already classifies their key points into different visual words. However, for new patterns, the knn classifier is needed.

The final step is to form feature vectors for image patterns by counting the occurrences of all key points in each image pattern. Given a new image of congested traffic, interest points found by Fast-Hessian detector are classified into typical key points using the knn classifier. Next, a histogram which counts the number of points in each of the typical groups is formed and conceived as the feature vector of given pattern. The bottom part of Fig 4.4 illustrates clearly these two steps. Eq. 4.3 formulates the point-based feature vector for a pattern of congested traffic. It can be seen that, the pre-specified number of typical key points is the dimension of the feature vector of congested patterns.

$$f_P = (n_1, n_2, \dots, n_K) \quad (4.3)$$

Where,

K is the number of visual words (groups of typical key points) in the visual word dictionary

n_i is the number of i -th visual word in the pattern

4.3.3 Area-based feature extraction

In contrast to the point-based method, the area-based approach explores features at higher abstract levels. The authors in (Krishnakumari et al., 2017), for instance, discern different congested patterns by employing the so-called Active Shape Model (Cootes et al., 1995) to represent different interested contours found in speed images. Although their approach is promising, the classification accuracy is limited due to the insufficiency of the naive contour extraction. Here, we extend this method further to overcome this shortcoming by using more sophisticated segmentation method to more accurately identify shapes of different (and relevant) elements in congestion patterns.

Furthermore, a more refined feature vector is also formulated to have a more descriptive representation of congestion patterns.

We propose three steps for extracting high-level custom features from a traffic pattern: (i) segmentation of congestion patterns, (ii) feature vector formulation which extracts traffic-related features from segmented patterns and then formulate a suitable feature vector. Fig. 4.5 represents the outline of the process. Each of the building blocks are detailed further below.

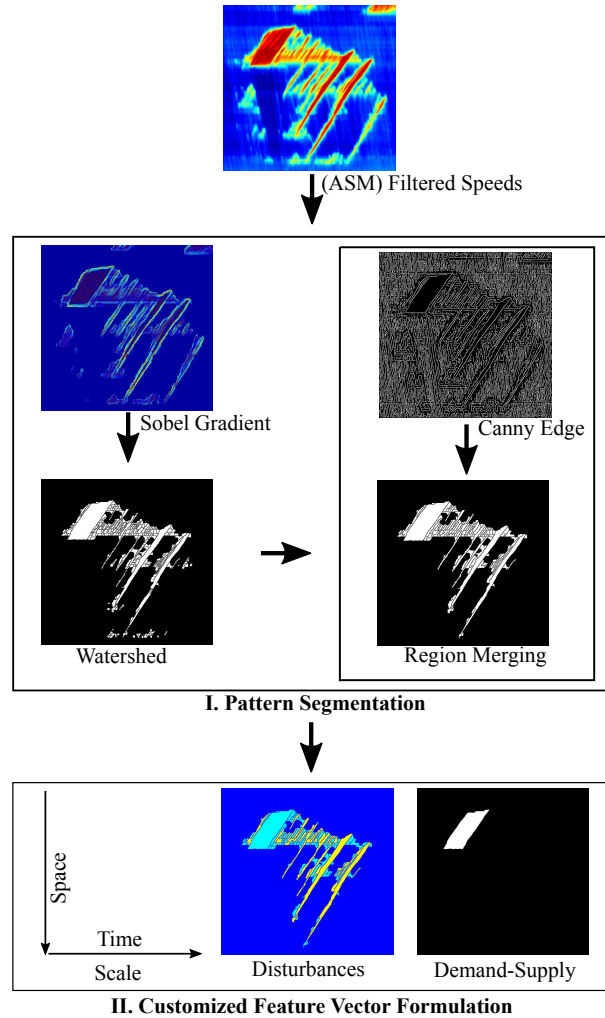


Figure 4.5: Area-based feature extraction using the segmentation approach

Pattern segmentation

One of the main reasons for false negative classification of the classifier in (Krishnakumari et al., 2017) was the contour/area detection from the traffic patterns. This segmentation step aims to divide image representation of traffic patterns into segments. Here, we use Watershed which is a sophisticated segmentation method in the field of image processing (Beucher & Meyer, 1992).

Watershed algorithm uses the gradient information of the original image. Gradients of speed images represent a space time map of (estimated) mean accelerations. The edges in speed images are detected and labelled with the maximum value of gradient. Watershed is then applied to the gradient image. This algorithm usually leads to over-segmentation and hence, a refine step - region growing - is further implemented. These steps are illustrated in Fig. 4.5 and detailed in the following sections.

Gradient image Gradient image describes how strong the (speed) value changes across spatio-temporal (pixel) location in the original speed map. It is calculated by Sobel method (Sobel, 1990), in which gradient is measured in both dimension of space and time. This is done by applying convolution of the speed map and appropriate Sobel operators as follows (t, x indicate temporal and spatial, respectively):

$$S_t = \begin{bmatrix} +1 & 0 & -1 \\ +2 & 0 & -2 \\ +1 & 0 & -1 \end{bmatrix}, S_x = \begin{bmatrix} +1 & +2 & +1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix}$$

The horizontal and vertical gradients (G_t, G_x) of the pattern, which has speed matrix indicated by $I(x, t)$, are calculated as shown in Eq. 4.4 (in which, \otimes indicates convolution operation).

$$\begin{aligned} G_t &= S_t \otimes I \\ G_x &= S_x \otimes I \end{aligned} \quad (4.4)$$

Eq. 4.5 combines these multicomponent gradients to yield the magnitude of the final gradient.

$$G = \sqrt{G_t^2 + G_x^2} \quad (4.5)$$

The last step is to normalize the gradient value.

$$G_0 = \frac{G}{\max(G)} \quad (4.6)$$

Watershed Watershed is one of the main morphological operators in the field of mathematical topology which has been applied in image segmentation (Beucher & Meyer, 1992). The basic concept of this method is to consider a gray-scale image as a topographic surface in which altitude of a point is set to its intensity value. Fig. 4.6 demonstrates the concept using a simple 2D signal. The surface is gradually immersed into water. Virtual holes at all minima can let the water rise through. When a minimum is reached by the water, a (catchment) basin is formed accordingly. As shown in Fig. 4.6a, water level will first reach the lower minimum, and then water will spread into the surrounding area. A blue basin is formed for this minimum. A moment later, water will then reach the second minimum and also form the green basin (see Fig. 4.6b).

Water level continue rising and when it reaches the local maximum, the water from two basins are about to meet. At this point, a dam, which is labelled by red colour, is built to prevent this as depicted in Fig. 4.6c. Dams will get higher as water level gets higher. This process continues until the surface is completely flooded. As a result, we will get various image segments, in different colours, separated by shed lines or dam as simply shown in the figure.

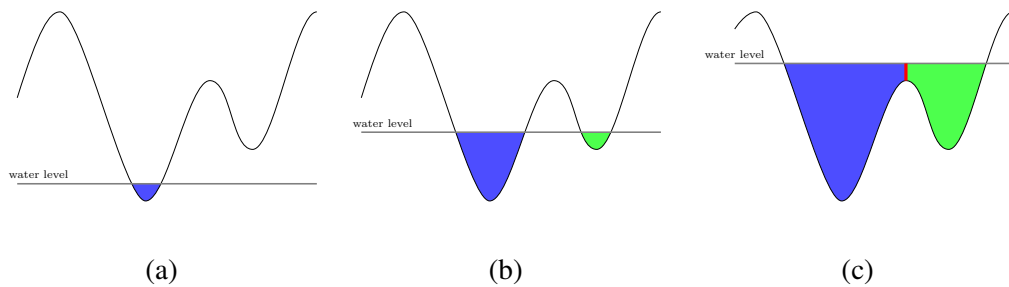


Figure 4.6: Demonstration of fundamental steps in Watershed segmentation. Water level reaches the first minimum (a), water level reaches the second minimum (b) and red dam prevents water merging (c).

Algorithm 4.1 presents the Hill climbing algorithm for Watershed segmentation (Rambabu & Chakrabarti, 2008). It illustrates precisely the principle of immersion/flooding process above, although it is derived from topological-based definition of Watershed transform. \mathcal{F} comprises the top part of the topographic surface that is above the water level and 1-pixel high part that is just reached by the water (In initial state, only the boundary of this part is included). The pixel x taken from step six has the lower intensity, i.e. the pixel that is just reached by the water level. The algorithm assumes water growing from this x to its neighbors x' . If x' has not been labeled yet, it will be assigned to the same basin with part of water that touched it, i.e. x 's label. On the other hand, if it is assigned to a different basin, water from two different basins meets here and hence, this should belong to the watershed line. The algorithm continues until all pixels are visited.

Region merging This step is to deal with the common over-segmentation consequence of watershed. The strategy is to merge a region with its neighbors, if reasonable, to obtain a larger region. The criteria to determine where to merge two neighbor regions is that their speed difference should not be too large. Consequently, edges should not be present between these two regions. Therefore, canny edge (Canny, 1986), an extremely well-known technique in image processing, is used as a criteria for region merging. In principle, it comprises of four main steps: (1) image is smoothed out by a Gaussian filter in order to remove noise, (2) Sobel gradient is calculated on the smoothed image, (3) applying non-maximum suppression to find local maximum intensities in gradient image which correspond to edge points, and (4) two upper-bound and lower-bound thresholds are used to filter out unwanted edge points.

Algorithm 4.1 Hill Climbing algorithm for Watershed transform**Require:**

Gray scale image represented by intensity function f

Set of regional minima $\mathcal{M} = \{m_i\}, i = 1 : M$ with value α_i respectively

Initialization

1: **for each** $m_i \in \mathcal{M}$ **do**

2: $\forall p \in m_i, \text{label}(p) \leftarrow i$

3: $\mathcal{S} \leftarrow \{p \in m_i | \forall p' \in N_{G(p)} : f(p') = \alpha_i\}$ /* All interior points of regional minima */

4: **end for**

5: $\overline{\mathcal{S}} \leftarrow \{p \notin \mathcal{S}\}$

Processing

6: **while** $\overline{\mathcal{S}} \neq \emptyset$ **do**

7: $p = \underset{p \in \overline{\mathcal{S}}}{\operatorname{argmin}} f(p)$ /* Take the lowest point */

8: $\overline{\mathcal{S}} \leftarrow \overline{\mathcal{S}} \setminus p$ /* Remove p from set $\overline{\mathcal{S}}$ */

9: **for each** $p' \in N_{G(p)} \cap \overline{\mathcal{S}}$ **do** /* neighbors of p in $\overline{\mathcal{S}}$ */

10: **if** $\text{label}(p') = \text{Unknown}$ **then**

11: $\text{label}(p') \leftarrow \text{label}(p)$

12: **else if** $\text{label}(p') \neq \text{label}(p)$ **then**

13: $\text{label}(p') \leftarrow \text{SHEDLINE}$

14: **end if**

15: **end for**

16: **end while**

The proposed algorithm for this merging is represented in pseudo-code, see Alg. 4.2. Given a region, the general idea is to repeat merging it with its appropriate neighbors. The common boundary can be extracted by using fundamental morphological operators such as dilation and erosion (Haralick et al., 1987). The former operator dilates a connected region in binary images with respect to given number of pixels, while the latter one does exactly the opposite. The dilation operator expands a region to surpass the shed line separating neighbor regions. Hence, the common boundary can be identified by the intersection of these expansion parts. Afterward, erosion operator shrinks the obtained boundary to its original size for further steps.

Algorithm 4.2 Region merging algorithm**Require:** \mathcal{R} : set of regions, \mathcal{E} : set of edges

```

1: for each  $r \in \mathcal{R}$  do
2:   repeat
3:      $\mathcal{N}_r \leftarrow \text{Neighbours}(r)$ 
4:      $\text{isNewMerge} = \text{false}$ 
5:     for each  $n_r \in \mathcal{N}_r$  do
6:        $c \leftarrow$  common boundary of  $n_r$  and  $r$ 
7:       if  $c \cap \mathcal{E} = \emptyset$  then
8:         merge  $n_r$  and  $r$ :  $r \leftarrow r \cup n_r$ 
9:          $\text{isNewMerge} = \text{true}$ 
10:      end if
11:    end for
12:  until  $\neg \text{isNewMerge}$ 
13: end for

```

Customized Feature Vector Formulation

To have a better view on which features we should extract from a pattern of congestion, we look further into the pattern in Fig. 4.5. This pattern is significantly notable in the sense that it encompasses different typical elements. A severe congestion occurred which is likely caused by an accident and results in homogeneously and heavily congested traffic - represented by a large region of red colour in the related speed image. For the sake of simplicity regarding name convention, we refer to this as (heavily congested) *demand-supply component*. Following up are numerous disturbances which spill back against driving direction. Being motivated by this, a two-level hierarchical definition of congestion pattern is formed, namely phenomenon and pattern levels. At the lower level, three important elements related to traffic flow domain are explored to describe a congestion pattern: space-time scale, disturbances, and demand-supply components. The first element approximates the extension of a congestion pattern by measuring its temporal and spatial extent. The second element is identified in (Krishnakumari et al., 2017) as the most common traffic phenomenon where both of its upstream and downstream heads move against traffic direction. The last element has downstream front be stationary for longer time spans and heavily congested traffic upstream. At the higher level, a congestion pattern can be conceived as a combination of lower-level elements. The area-based approach aims to identify this higher-level congestion pattern. Following sections describe those lower-level elements in further details.

Congestion scale The congestion scale encompasses two measurements: spatial and temporal extents of congestion. The former is the total length of the road stretch being reached by traffic congestion. The latter indicates approximately how long the

congestion lasts. For each location, the duration for which congestion has occurred is calculated. The highest value (over space) is used as the representative duration for the temporal extent. An illustration is shown in Fig. 4.5.

Disturbances identification Disturbances are observed in traffic very often. They occur either as a small disturbance inside synchronised traffic near a bottleneck or as a wide moving jam. Krishnakumari et al. (2017) successfully proposed using Active Shape Model to identify WMJs. Since small disturbances and wide moving jams have similar shapes except for their sizes, this chapter further employs this method to determine disturbances in congestion patterns.

The Active Shape Model technique (Cootes et al., 1995) describes a shape using a mean shape and its variations derived from a set of similar shapes. Thus, given a new shape, the error while fitting the shape model to this shape can be used for identifying/classifying the shape. Krishnakumari et al. (2017) gives a detailed explanation of how to build a shape model of WMJ and how it can be used as a shape classifier. For this work, we only consider the WMJ shape, so the feature vector for classifying a given shape slightly varies to include (i) fitting error to WMJ shape model, (ii) affine transformation parameters of the fitted shape, (iii) time extent of WMJ shape, and (iv) space extent of WMJ shape. A simple logistic classifier is used on this feature vector to identify whether a shape is a WMJ (meaning disturbances in our case).

Demand-Supply identification In contrast to disturbances, demand-supply (DS) related elements commonly have downstream head being fixed at a location. In this work, we defined noticeable DS elements with following criteria: (1) average speed is equal or lower than 30 km/h, and (2) downstream head is stationary for at least 15 minutes. This time period is chosen arbitrarily and is certainly changed according to application interest. In general, the longer this time is, the longer congestion is experienced at the bottleneck; hence severer the situation is. They are formulated as in Eq. 4.7. Due to the existence of noise in traffic data, the second requirement is relaxed as having congested traffic for at least 15 minutes at downstream part of the region. This means the location of long lasting congestion could be at any location downstream of a congested region. We have taken 30 percent of corresponding (congested) road stretch as the searching area for this location.

$$isDS(r) = \begin{cases} 1 & \mu_{speed(r)} \leq 30 \text{ km/h and} \\ & congested_time(\text{downstream head}) \geq 15 \text{ minutes} \\ 0 & \text{otherwise} \end{cases} \quad (4.7)$$

Where,

r is a congested region

μ denotes average

Feature Vector Formulation Based on these three feature components, we construct a feature vector for representing a traffic pattern as follows:

$$f_A = (l_{Space}, l_{Time}, n_{Disturbances}, isDS) \quad (4.8)$$

Where,

l_{Space} is the spatial extent of the congestion

l_{Time} is the temporal extent of the congestion

$n_{Disturbances}$ is the number of disturbance instances

$isDS$ indicates if any Demand-Supply elements exist in the congestion

In order to equalise the degree of influence of each feature dimension on the clustering decision, we normalised both the feature vectors from the two feature schemes f_P and f_A to the range of (0,1).

4.3.4 Synthesis

This section has described in depth three fundamental components which constitutes the whole procedure required to cluster a given dataset of congestions. The preprocessing component uses ASM to turn traffic data in spatiotemporal dimension into corresponding image representation, which enables the application of computer vision techniques upon. Furthermore, two feature extraction approaches, i.e. point-based and area-based, explore various types of features that are special local points or domain knowledge characteristics. Watershed segmentation is proposed as a method for segmenting a congestion (image) pattern into different areas. Finally, hierarchical clustering is used to analyse the structure of the given set of congestions. In the following sections, the proposed framework is applied to a case study and performances of different methods are assessed and compared.

4.4 Evaluation Methodology

The methodology for evaluating the different elements of the framework is designed and presented in this section. Firstly, the dataset used for applying the proposed clustering methodology is introduced. Then, comprehensive methods are described to evaluate the two main elements of the proposed framework - (i) Watershed segmentation performance, and (ii) clustering results from the two feature schemes. Watershed segmentation will be evaluated for its ability to divide a congested pattern into different spatio-temporal areas so that the domain-knowledge characteristics are derived. Similarity of intra-class patterns and dissimilarity of inter-class patterns are interpreted to evaluate the clustering results.

4.4.1 Dataset

In this study, we use speed data from the loop detectors as input for testing the two approaches. This data is provided by National Data Warehouse (NDW) ndw, which is an organization managing a large amount of traffic data from most of the Dutch highways. We chose two busy roadways - A12 and A13 a12; a13 - for our case study. The data is collected for four months: from April to July in 2016 (This temporal period is chosen arbitrarily). Congestion patterns are searched over every single day via the following steps. First, a smoothing method - the Adaptive Smoothing Method - is applied to obtain an image representation (a speed matrix) of traffic state during that day. Parameters of the filter are set based on the settings in (Schreiter et al., 2010a); some basic ones are shown in Table 4.1. Second, a speed threshold of 80 km/h is used to filter congested pixels in the image. Then, relevant image morphological operators such as opening or dilating are applied to remove noises and connect nearby congestion-related pixels, which have speed values lower than the speed threshold. This results in connected spatiotemporal regions of congested traffic. Bounding boxes of these connected regions are generated to extract congested patterns thereafter. Applying this for the A12 raw dataset results in 232 congested traffic patterns which constitute a primary dataset for our analysis. The two proposed methods will be evaluated on this A12 dataset. In addition, we obtained 196 patterns on A13 which is used as a secondary dataset for further evaluation as shown in Sec. 4.5.3 - Qualitative. Table 4.1 summarizes some details related to the parameters used in this study.

Table 4.1: Parameters and a few descriptions of the traffic patterns used in this study (A12 and A13 highway, the Netherlands, from April to July, 2016).

Adaptive Smoothing Method	Temporal resolution	0.5 minutes
	Spatial resolution	100 m
	Traffic wave speed	-18 km/h
Congested patterns extraction	Congested speed threshold	80 km/h
A12 dataset (primary)	Number of patterns	232
	Time range	10 – 320 minutes
	Space range	2.4 – 40.4 km
A13 dataset (secondary)	Number of patterns	196
	Time range	10 – 248 minutes
	Space range	2.4 – 18.8 km

4.4.2 Watershed segmentation evaluation

Watershed segmentation is the core component in the area-based feature scheme; it divides a congestion pattern into different spatio-temporal areas, particularly to separate

disturbances. Since having a ground truth set is not trivial, we visually assess the segmented patterns. An important qualitative criterion is a clear separation of disturbances in traffic congestions. Various examples with multiple levels of complexity are chosen to evaluate the Watershed segmentation results. In addition, since one of the main reasons for adopting this technique is to overcome the shortcoming of previous contour extraction, a qualitative comparison between these two techniques is conducted.

4.4.3 Clustering evaluation

Since cluster analysis is an unsupervised learning problem, an appropriate validation is not trivial. However, there are generally two validation criteria in the literature: internal and external (Rendón et al., 2011; Jain, 2010). An internal criterion validates a clustering result based on properties inherent to the given dataset. In contrast, an external criterion matches a clustering result with prior information on structure of the dataset, which is generally referred to as *true labels*. Such information is, however, normally either subjective or unavailable. Therefore, in this work, we focus on internal criteria to evaluate the clustering results. Our evaluation metrics comprises of qualitative and quantitative measures. The former examines patterns from each of the resulting clusters by visually assessing their similarity. The latter evaluates how well clusters are separated from each other by using quantifying the confidence level of appropriate classifiers trained on the corresponding clustered data.

Qualitative evaluation

The qualitative evaluation step assesses the *similarity* of patterns in each of resulting clusters. There is no quantitative definition of similarity, instead we base our assessment on the appearances of congestion patterns over space and time, equivalently to compare images of these patterns. Attempts to draw some common characteristics on these patterns are driven by traffic knowledge, which has received significant attention in the literature such as (temporal, spatial) scale of congestion or disturbances.

Quantitative evaluation

For quantitative evaluation, we analyse the *separability* of resulting clusters. This is motivated by the follow-up application in which new traffic patterns are classified preferably without repeating the clustering for the whole dataset (previous and new patterns). Hence, resulting clusters are used as a training set, on which a suitable classifier is trained to classify new patterns. The separability criterion is then measured by the *confidence level* of this classifier's decisions to assign patterns to clusters. It is generally expected that well separated clusters would confuse the classifier less, meaning higher confidence in classification.

To quantify the confidence level of a classifying decision, we adopt the family of classifiers which gives the membership probability that a pattern belongs to a class. The higher the probability, the stronger the classifier believes in its decision. Since the number of clusters can be higher than 2, we have a multi-class classification problem. In this work, we choose multinomial logistic classifier as our base classifier for evaluating the clustering results. A brief description of this method is provided later in this section.

Alg. 4.3 summarizes the procedure for quantitative evaluation of clustering results. K-fold cross validation is used to obtain a stable result. It starts by dividing the whole dataset into a number of groups (step 1). In each run, one fold is taken as a test set and the rest are combined into a training set. We perform cluster analysis on this training set to obtain clusters and subsequently class labels for its patterns (step 2). Next, a multinomial logistic classifier (\mathcal{F}) is trained on these samples (step 3) and then applied on the test set (step 4). \mathcal{F} provides membership probabilities of a given pattern to all classes. The class associated with the highest probability is chosen to label the pattern. This probability also shows the confidence level of the classifier. We apply \mathcal{F} to all patterns in the test set and collect all confidence levels (step 5) to evaluate the separability of the clustering results. In particular, The distribution of these probabilities are subsequently constructed for different number of clusters. We compare these for the two feature schemes to achieve a quantitative evaluation of clustering results.

Algorithm 4.3 Quantitative evaluation scheme of a clustering result

Initialization

1: Randomly divide the dataset into N folds (groups): $\mathcal{D}_i, i = 1 : N$

Evaluation

2: **for each** fold \mathcal{D}_i **do**

3: Cluster the set of other folds $\{\mathcal{D}_j, j \neq i\}$

4: Learn a multinomial logistic classifier, \mathcal{F}_i , given the obtained clusters as a training set

5: Fit \mathcal{F}_i on every pattern in \mathcal{D}_i . Collect classifier's confidence levels which are the probabilities that \mathcal{F}_i provides to support its decisions to classify patterns (to one of the obtained clusters)

6: **end for**

Multinomial logistic regression The multinomial logistic regression (MLR) (Böhning, 1992) is an extension of the well-known binomial logistic regression that deals with multiclass classification problems. A softmax function is used instead of a logit function to calculate the membership probability that an observation (or pattern) belongs to one particular group. Thus, it is also called softmax regression. The predicted probability of i th category is calculated by Eq. 4.9, given a feature vector x and a coefficient matrix W . The input for the softmax function is a linear combination of feature vector and coefficients of class i , which is the corresponding row in W .

$$P(y = C_i|x, W) = \frac{e^{W_{C_i}^T x}}{\sum_{j=1}^C e^{W_{C_j}^T x}} \quad (4.9)$$

The coefficient matrix W can be optimized using different methods to fit the MLR model (Huang et al., 2010; Yu et al., 2011; Gao et al., 2007). In this work, we use the dual coordinate descent method for the optimization (Yu et al., 2011).

4.5 Results and Discussion

This section presents and discusses the Watershed segmentation and clustering results of the two feature schemes in the proposed framework. Details are presented in the following sections.

4.5.1 Size of the (visual) word dictionary

The number of key points in a dictionary is represented by K (see Eq. 4.3), which is also a parameter of the applied k-means clustering. To determine an appropriate value for K , we minimize the ratio between intra-cluster (SSE - sum of squared errors) and inter-cluster (distance to the closet cluster) distances (defined in Eq. 4.10) as proposed by Ray & Turi (1999).

$$\begin{aligned} D^{intra} (= SSE) &= \sum_k \sum_i (x_i^{C_k} - c_k)^2 \\ D^{inter} &= \min_{p \neq q} (c_p - c_q)^2 \\ c_k &= \frac{1}{|C_k|} \sum_i x_i^{C_k} \end{aligned} \quad (4.10)$$

in which, C_k is the k _th cluster and has $|C_k|$ instances, $x_i^{C_k}$ is the i _th instance of C_k , and c_k is the centroid of C_k . In our case, the minimisation of the ratio resulted in $K = 16$.

4.5.2 Watershed segmentation of congested patterns

As described in the evaluation methodology, we visually analyze the performance of Watershed segmentation (WS) on a variety of congested patterns. Additionally, a qualitative comparison with contour extraction from (Krishnakumari et al., 2017) is provided. Fig. 4.7a shows examples of congestions from the dataset with different complexity, such as congestion with a single isolated disturbance, multiple sparse disturbances in a congested area and finally, a highly dense congested area. Generally, segmenting these patterns becomes harder as the complexity becomes higher due to

the spacing between the disturbances becoming less evident. Corresponding WS results are shown in Fig. 4.7b. Different regions in a pattern are separated by shed lines, which are indicated in dark blue. The region colours are mapped from the (arithmetic) mean speed of all the interior pixels of a segment. Free flow regions are coloured with the same colour as the shed line since they are referred to as uninterested regions in this study.

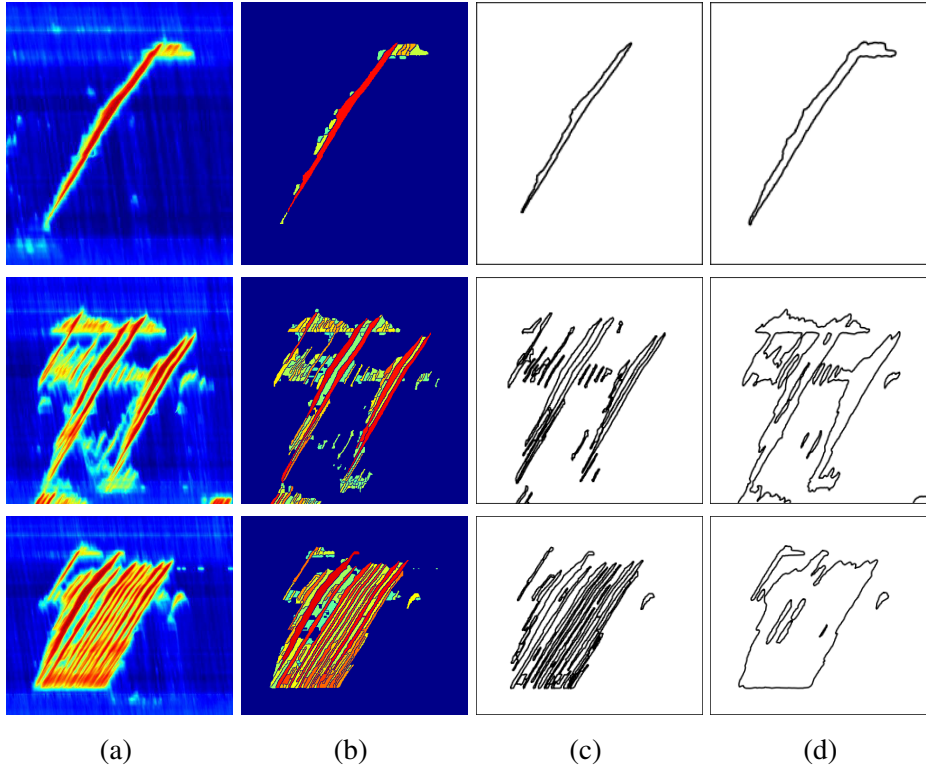


Figure 4.7: Examples of segmenting congested patterns into different areas. The left column (a) are some various congestions in spatio-temporal representations. Results of the Watershed segmentation is shown in column (b). The corresponding contours extracted from the Watershed-based segments and the naive contour extraction are in column (c) and (d) respectively.

We can see that the WS follows and groups pixels in moving jams. Since there is no ground truth for the segmentation, we do not assess the accuracy at the pixel level, but assess the ability of the WS to give distinct disturbances or wide moving jams. In the first pattern in Fig. 4.7a, WS performs satisfactorily as it can separate the prominent pattern, i.e. the moving disturbance, from the other regions. In the second congestion pattern, several disturbances occur with various sizes and most of them are relatively far apart from each other. The WS result shows that noticeable disturbances are separated clearly. The third pattern is a highly dense congestion with many moving jams being extremely close to each other. Their heads are mixed with the upstream bottleneck which is the potential reason for their emerging. The corresponding WS result presents acceptable segmented regions. Moving jams are tracked and separated correctly even though they are sometimes merged together as shown in the third pattern in

Fig. 4.7b. It is clear that tracking and separating these moving jams manually is also difficult. To summarize, segmentation results obtained from WS method show a high ability of separating disturbances in patterns of congestion.

The contour extraction from (Krishnakumari et al., 2017), also termed as naive contour extraction in this chapter, is also applied to these patterns to provide a comparison. The main criterion is how well they support the disturbance identification step. The corresponding contour extraction results are shown in Fig. 4.7d. For the WS, the contours are obtained by extracting the outlines of the corresponding regions as shown in Fig. 4.7c. It can be observed that, for the simple pattern with one single disturbance, the first one in Fig. 4.7a, contours obtained from the two methods are able to identify the corresponding disturbance. However, for more complex patterns, naive contour extraction is unable to identify different contours of moving jams. Hence, WS-based contour extraction performs superior to naive contour extraction method and can improve the quality of detecting disturbances.

In summary, some insights can be deduced from the performance of the WS as follows:

- Watershed segmentation is capable of tracking the propagation of a disturbance. It also works well for densely congested areas where multiple moving jams are close to each other.
- Contours extracted based on WS segmentation appear to capture moving disturbances better than that of naive contour extraction.

Since Watershed relies on the gradients in image patterns, its performance in recognising disturbances is likely affected by the choices of ASM parameters. Nonetheless, our analysis suggests that this method is quite robust with respect to ASM parameters. We refer readers to Appendix A for the complete analysis.

4.5.3 Clustering of congested patterns

This section presents the clustering results corresponding to the two feature sets described in the framework - point-based and area-based. They are evaluated using the methodology described in Section 4.4.3. The hierarchical agglomerative clustering technique results in a hierarchy of data points and is effectively represented by dendrograms, which are binary tree-based representations. The height at which two branches are combined denotes their distance or dissimilarity. Fig. 4.8 shows two dendrogram plots of the clustering results using the point-based and area-based features respectively.

Upon initial inspection, it can be seen that these two trees exhibit strikingly different properties in at least two perspectives: (i) leaf- combinations, (ii) inter-cluster dissimilarities. Firstly, leaf combinations show how similar any pairs of patterns (i.e. leaves) are. In the point-based tree, most of the patterns connect to fairly long vertical stems;

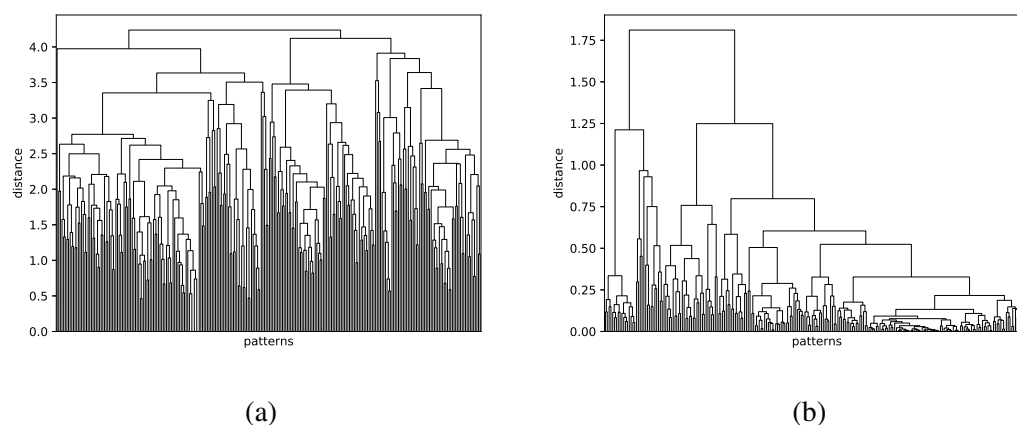


Figure 4.8: Dendrogram representations of hierarchical clustering results from different feature schemes: (a) Point-based features and (b) Area-based features. On the horizontal axis lie all the patterns in the dataset. For a clear visualization, the dendrogram organizes all patterns in a way that allows patterns in the same cluster (at any iteration of the hierarchical clustering algorithm) stand next to each other and form a group. (Notice that, there are different orders that satisfy the requirement). The vertical axis shows distances measured between two clusters. One can observe the dendrogram from bottom to top and see when (the order) and which two patterns/clusters are grouped into a bigger cluster.

this indicates certain levels of dissimilarities between these patterns. On the other hand, the majority of the (direct) leaf connections in the area-based dendrogram are through extremely short vertical stems, meaning strong similarities between them. (Intuitively, one can easily observe this by looking at the dense parts at the bottom of each of the two dendrograms.) Secondly, regarding (intermediate) cluster differences, the two dendrograms show different properties. One can look at the upper vertical stems, which are not directly connecting leaves, to analyse these inter-cluster dissimilarities. While in the left dendrogram, the lengths of these stems hardly get longer from bottom to top (as clusters are becoming bigger), the right dendrogram shows significant increases of the lengths of these stems. This suggests a higher level of difference between patterns representing by the area-based feature scheme compared to that the point-based scheme. There are two possible hypotheses for these differences in these two dendrograms: the dimensionality of feature vector and the characteristics conveyed by extracted features. In fact, one can expect that the higher the dimension of a feature vector is, the further the distances between patterns is likely to be. In addition, traffic engineers consider phenomena happening in patterns to judge if they are similar. To some extent, this is inline with the approach of area-based method which looks at patterns at a global, abstract level. The point-based looks into more local features which can possibly find differences between *similar* patterns. Although this does not imply a better clustering of the area-based approach, this explains the correlation of feature characteristics and resulting dendrograms.

Concerning the choice of the number of potential clusters in the dataset, one can cut a dendrogram at a certain horizontal level. From these, separated sub-trees are formed, each with an individual cluster. It is not clear from the point-based dendrogram where to cut the tree since the inter-cluster dissimilarities are mostly much smaller than that of intra-cluster. In contrast, the area-based tree does suggest some potential number of clusters. For instance, we can cut the tree in fig. 4.8b by a horizontal line at distance 1.00 to obtain 4 different clusters. The following sections evaluate these clustering results using the proposed qualitative and quantitative methods.

Qualitative

To assess the quality of the clustering results, we consider different (intermediate) clusters, i.e. branches, suggested by the two dendrograms in Fig. 4.8. The quality of a clustering is measured based on similarities between patterns in the same cluster. Here we visually analyse patterns with respect to their spatio-temporal appearances and traffic knowledge as described in the evaluation methodology.

For the point-based dendrogram, we first look at the two topmost sub-trees (or branches) and see that: While the left sub-tree consists of small, simple patterns of disturbances, the right sub-tree observes a variety of patterns. Therefore, we further look at clusters that (i) are at low distances, meaning high similarities between patterns, and (ii) have a relatively high number of patterns, for instance, clusters with less than ten objects that are not considered. By doing so, we found five types of traffic congestion. Some examples of those are shown in Table 4.2. Their locations in the dendrogram are highlighted in Fig. 4.9.

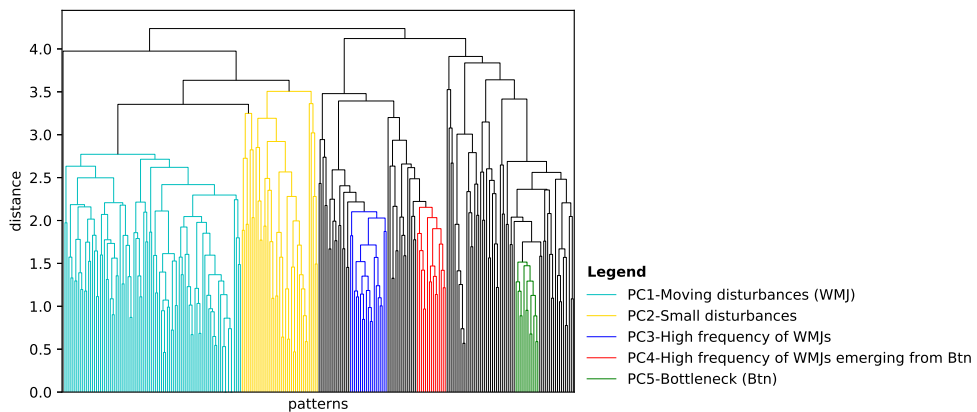
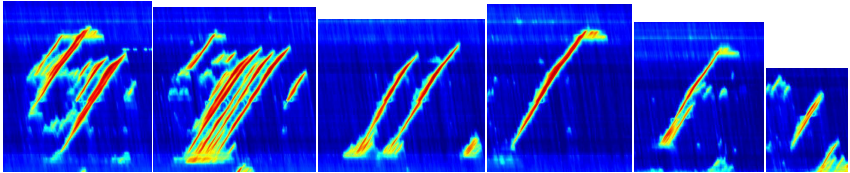
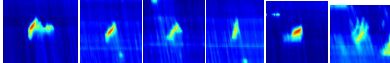
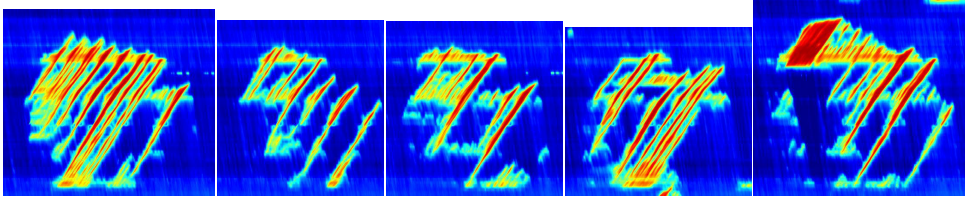
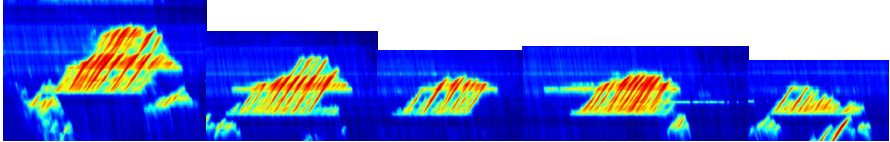
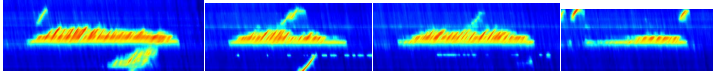


Figure 4.9: Visualization of four typical congestion patterns found in the dataset by applying hierarchical agglomerative clustering on the point-based feature scheme

It can be seen from Table 4.2 that this approach is capable of capturing some typical patterns of congestion. The left (topmost) sub-tree encompasses two clusters of patterns, namely *Moving disturbances (WMJ)* - PC1 and *Small disturbances* - PC2. Most of the patterns in PC1 present single or a very few disturbances spilling back against the

driving direction, which are so called wide moving jam (WMJ). The PC2 comprises of very light disturbances that are both spatially and temporally short. In cluster PC3 and PC4, there are many moving disturbances in various combinations. The striking difference between these two is that those in PC4 emerge from fixed bottleneck locations while that is not the case in PC3 where origins of disturbances are at different locations. Therefore, to discern between these patterns, we call them *High frequency of WMJs* and *High frequency of WMJs emerging from bottlenecks*, respectively. The last cluster (PC5) shows traffic congestion that is stationary at fix bottlenecks for long periods; hence, we name it *Bottleneck*.

Table 4.2: Description of point-based clustering result. These clusters are corresponding with clusters shown in the dendrogram in Fig. 4.9. It should be noticed that they are represented in a deliberate order to facilitate the corresponding discussion.

Cluster	Number of patterns	Examples
PC1	90	
PC2	39	
PC3	19	
PC4	15	
PC5	13	

Despite the four typical patterns that have just been described, there are some limitations of this approach as follows. Firstly, the clusters listed in Table 4.2 are sub-trees of the corresponding dendrogram (as highlighted in Fig. 4.9), which in aggregation does not cover the whole dataset. It takes much effort, e.g. trial-and-error, to explore and locate them in the dendrogram. Secondly, from a traffic knowledge point of view, similar patterns are expected to be grouped at as much lower level (of distances). This requirement is not met with the point-based results. For instance, the PC2 consists of two (yellow) sub-trees which are not directly connected with each other even though they describe the same pattern. Moreover, there are many patterns such as WMJs and small disturbances in black colour that are not partitioned to their corresponding PC1

and PC2. Their unexpected positions are rather hard to explain; one might need to investigate their extracted key points to do so. As shown previously, key points essentially captured characteristics of neighbourhoods surrounding particular locations in images. Additionally, the construction of a dictionary of visual words disregards potential spatial or temporal correlations of these points and only considers their occurrence frequencies. That means, to some extent, losing potentially valuable information is inevitable.

For the area-based approach, we also examine patterns from various (intermediate) clusters. The obtained result include five clusters with strong intra-cluster similarities. Table 4.3 presents some examples from these clusters together with their sizes. Additionally, their corresponding locations are highlighted in the dendrogram in Fig. 4.10. It can be seen from the dendrogram that these clusters encompass the whole dataset. Statistically, the smallest cluster - AC5, has 15 patterns while most of the patterns, particularly 154, are categorized into the first cluster (AC1). This means AC1 is the most recurrent congestion found in the A12 road segment for the four studied months.

The first cluster (AC1) shows congestion with one or a very few disturbances. Most of them describe so-called wide moving jam with the minor are insignificant disturbances, see examples in Table 4.3. We therefore name this cluster *disturbance*. The second cluster (AC2) shows patterns with a number of moving disturbances. Consequently, a reasonable label for this cluster is *high frequency of disturbances*. It is noticeable that most of the patterns in the first two clusters constitute of moving disturbances. The third cluster (AC3) illustrates *bottleneck* congestion as described in the point-based approach. Patterns in cluster AC4 are small scaled both spatially and temporally compared to those in other clusters. Furthermore, they mainly show the demand-supply patterns, which cause slow traffic. Since traffic presented inside these patterns is homogeneous, a suitable label for this cluster is *small scaled homogeneous congested traffic*. Lastly, the fifth cluster (AC5) shows complex patterns with compounds of homogeneous congestions and moving disturbances. Additionally, their spatio-temporal scale is large. Hence, these are defined as *Homogeneous & Disturbances* clusters.

The dendrogram in Fig. 4.10 can lead to some insights into similarity scores within each of the five clusters. Cluster AC4, in blue, has the lowest distance, meaning lowest variance. The construction of the feature vector is likely to account for this minimal intra-cluster dissimilarity. Firstly, the demand-supply elements are logically indicated as existing in these patterns, regardless of its properties (such as shape or scale). Secondly, there are very few, if not no, disturbances in these patterns. Thirdly, the congestion is in small scale. These three characteristics lead to a minimal distance between patterns in cluster AC4. Intra-cluster distances of AC3 and AC4 are more or less at the same level. AC2 is a bit higher in comparison and AC5 is the most diverse one. The complexity of *Homogeneous & Disturbance* patterns - large differences in scales and various number of disturbances (see examples in Table 4.3) - appears to account for this highest distance in AC5.

It is noteworthy that obtaining these five clusters by cutting the dendrogram with one

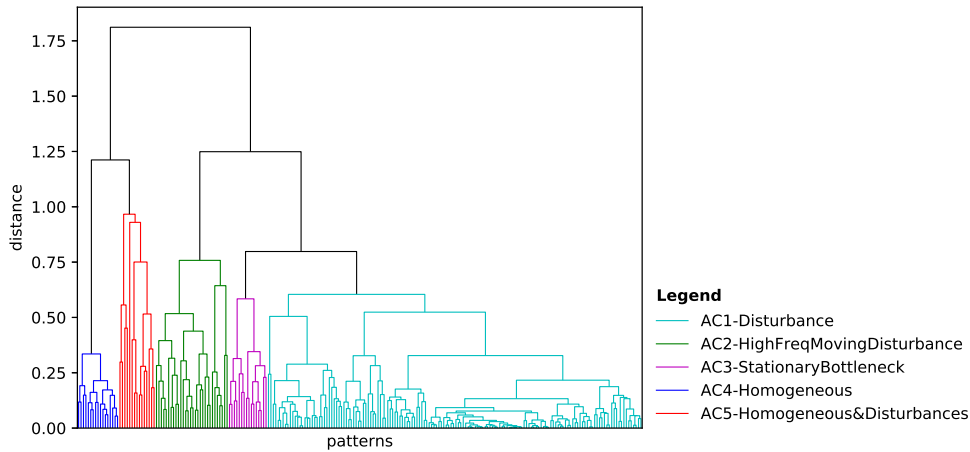
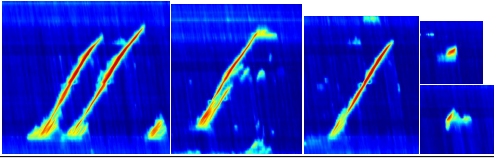
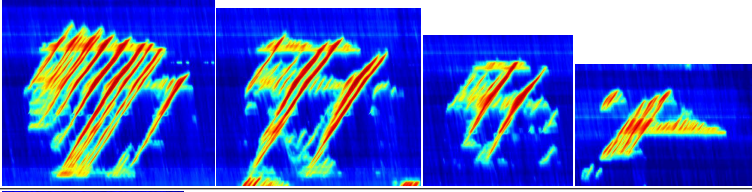
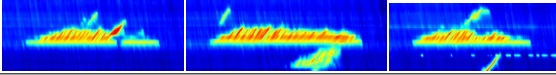
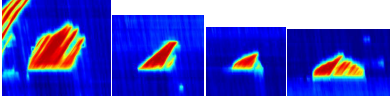
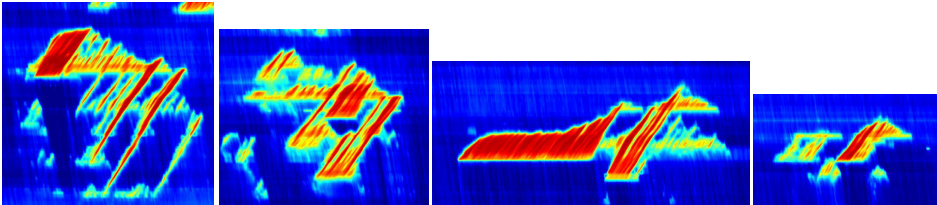


Figure 4.10: Visualization of five typical congestion patterns found in the A12 dataset by applying hierarchical agglomerative clustering on the area-based feature scheme

horizontal line is not possible. This situation is caused by high levels of dissimilarity of patterns in the compound pattern cluster. To automate the process of getting these five clusters, one can define a hierarchical approach as follows. (1) The first step is to cut the dendrogram into two clusters, which are the two branches with and without demand-supply elements in congestion patterns. Let us call them CisDS and CnoDS respectively. (2) The second step deals with these clusters separately. The CisDS is cut into two branches, which are AC4 and AC5. The CnoDS is cut into three branches which are AC1, AC2, AC3. An alternative approach can be a manipulation of the distance function or feature weighting to control the distance between AC5 patterns. However, we do not explore this further in this study.

A comparison between these two clustering results can be made from two perspectives. First, regarding classification application, the area-based approach successfully groups patterns into corresponding categories as its five clusters cover the whole dataset. In contrast, the four clusters from the point-based approach only capture a part of the dataset. Second, from a viewpoint of typical pattern recognition, both approaches show clusters of *bottlenecks* and *WMJs*. We ignore the fact that the point-based approach can miss some of these patterns and rather focus on the typical emerging patterns. While the area-based approach groups all the High frequency of WMJs together, the point-based approach can separate them into those that originate from bottlenecks and those that do not. The area-based approach can identify and separate demand-supply patterns into simple ones and compound ones that are most likely to be omitted by the point-based approach. This comparison suggests that the area-based approach is applicable for the classification of congestion patterns at a high abstraction level, and the point-based approach tends to work well with a lower detail level. This is well aligned with the methodology adopted in these two approaches. Hence, one can hierarchically start with the area-based approach to obtain the five clusters and use the point-based approach to partition the High frequency of WMJs further, for instance.

Table 4.3: Description of area-based clustering result. These clusters are corresponding with clusters shown in the dendrogram in Fig. 4.10. They are ordered deliberately to facilitate the corresponding discussion

Cluster	Number of patterns	Examples
AC1	154	
AC2	30	
AC3	16	
AC4	17	
AC5	15	

Interestingly, the five clusters resulting from the area-based method coincide largely with the five manually selected labels in previous work (Krishnakumari et al., 2017; Nguyen et al., 2016) (as mentioned in Section 4.2). It can be seen that these classes are well captured with the clusters in Table 4.3. The first two classes (WMJ + low frequency of WMJs) are generally grouped in cluster one, high frequency of WMJs in cluster two, homogeneous congestion in cluster four and mixed class cluster in cluster five. It should be noted that dividing traffic oscillation into low and high frequency is somewhat subjective due to the decision of threshold number of moving jams. There is a difference to the approach described here; stationary bottleneck patterns are clustered by the area-based method which is not considered in (Krishnakumari et al., 2017; Nguyen et al., 2016).

The area-based clustering results are also in line with the classification in (Helbing et al., 2009) that we discussed in Section 4.2. Our C1 cluster matches with the moving cluster C1. Both SGW and OCT comprise a frequent occurrence of moving jams, which can also be represented by cluster AC2. As noticed by the authors in (Helbing et al., 2009), differences between these two types are not straightforward. However, one might suggest to depend on the (temporal) distances between moving jams to discern

them. Subsequently, introducing further variables to the feature vector such as density of moving jams over time, might separate AC2 into SGW and OCT. PLC is essentially recognized by cluster AC3 and HCT by AC4. The WSP is not recognized in this result. This is likely due to the choice of the low-speed threshold, which might discard these patterns.

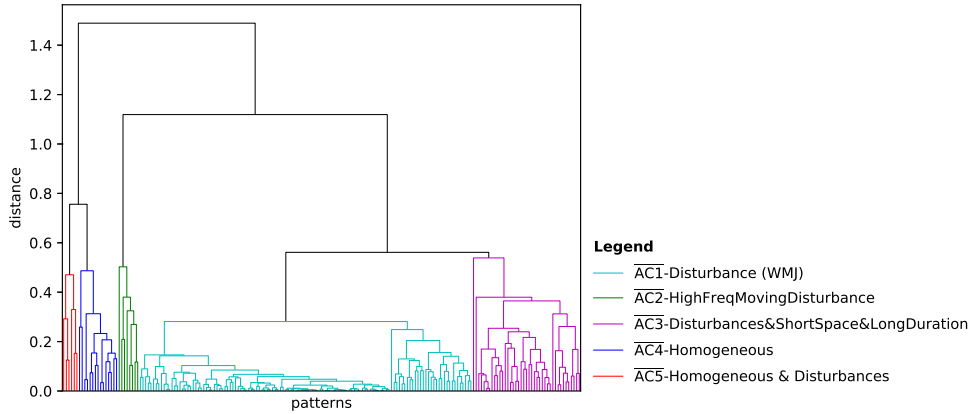


Figure 4.11: Visualization of five typical congestion patterns found in the A13 dataset by applying hierarchical agglomerative clustering on the area-based feature scheme.

In order to test the transferability of the area-based method, we apply it to clustering of congestion patterns in the A13 dataset. The resulting dendrogram has a similar structure to that obtained from A12 dataset, from which five clusters are identified and highlighted, as shown in Fig. 4.11. Four of them are comparable with those found in A12 dataset: disturbances, high frequencies of disturbances, homogeneous and mix of homogeneous and disturbances. The cluster $\overline{AC3}$ consists of bottleneck related patterns like those in cluster AC3 (A12 dataset); however, it also includes patterns that have many disturbances but with spatially short extents. Hence, it appears that short space, long duration and having disturbances are the common features of patterns in $\overline{AC3}$. Some of these patterns can belong to $\overline{AC2}$. This is possibly because that although the area-based feature vector includes disturbances and scales of patterns, it ignores the spatial length of such disturbances. One might include this information (disturbance lengths), in an appropriate approach, to distinguish spatially short disturbances in bottlenecks with long moving disturbances in $\overline{AC2}$. Nonetheless, the result from A13 dataset indicates that the area-based approach is transferable and able to cluster a dataset into meaningful groups.

Quantitative

We apply the evaluation methodology described in Section 4.4.3 with the following settings: 100 folds ($N = 100$) of the dataset and number of clusters varying from three to eight. We collect every confidence level of the classifier when assigning class labels to patterns; those levels, calculated by the classifier, are probabilities associating with

corresponding labels. Box plots are used to present the distribution of these levels of confidence in each option of the number of clusters. Fig. 4.12 shows the final result.

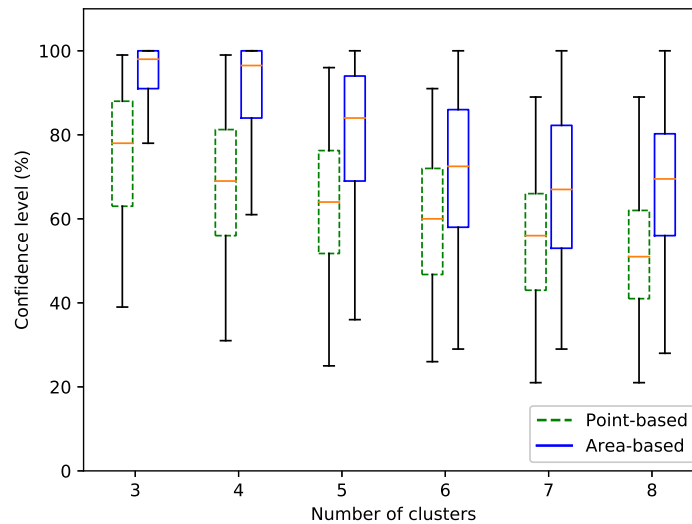


Figure 4.12: Quantitative evaluation results of the area-based and point-based approaches. Each box plot includes following elements: short horizontal bars, so-called whisker, at top and bottom extremes indicate maximum and minimum values respectively; the lower bound of each rectangular is at 25 percentile and the upper bound at 75 percentile; the horizontal red line inside the rectangle of each box plot indicates the median value.

Overall, a general declining trend is observed for both methods. This tendency is generally expected since the problem becomes more challenging when the dataset is partitioned into a higher number of clusters. In fact, the corresponding classifier becomes less confident in its decisions when more classes are involved. Additionally, given the same features, the separability of clustering decreases when the number of clusters increase. This leads to the decrease in the confidence level of the classifier. Fig. 4.12 also shows that the result of the area-based approach is superior to that of the point-based approach because of the higher locations of the rectangle boxes, which encapsulate the central 50% population of confidence values. Additionally, the median confidence levels of the area-based scheme are considerably higher than that of the point-based scheme over all the choices of the number of clusters. The highest median confidence level observed with the point-based approach is approximately 80 with 3 clusters. Additionally, the confidence level drops to around 50% for larger number of clusters, which indicates a high confusion level of the classifier. On the other hand, all the median confidence levels in the area-based approach are higher than 65%. Three of them are higher than 80%, hence, higher than the highest value in the other approach. This indicates that the confidence of classifiers learned from area-based clustering is strong compared to the point-based. Hence, it can be concluded that the area-based

approach results in a higher level of separability of the data set than the point-based approach.

4.6 Conclusions

This chapter explores two methods to automatically label and separate traffic patterns that come in the form of images (heatmaps of speeds over space and time along a road corridor). Both methods have their origins in image processing and conceive spatio-temporal traffic patterns as images where each pixel represents speed in a particular space-time area. The point-based (bag-of-feature) approach explores key points - locations - in traffic (image) patterns. These points do not necessarily have a direct meaning related to traffic. The area-based method uses the Watershed operator to segment a pattern into different areas from which three main traffic related elements are extracted—spatial and temporal scales, disturbances and (heavily congested) demand-supply areas. In this respect, this second *new* approach is a hybrid method that combines image processing and (traffic) domain knowledge. Hierarchical agglomerative clustering is performed on the feature sets derived from both methods which results in different hierarchies of the data set.

Since labelling traffic (congestion) patterns is highly subjective, we test the methods both *qualitatively*, (does the method result in meaningful clusters / labels?) and *quantitatively* (does the method separate the resulting feature space such that we can have high confidence in the results when applying a classifier using the labels on (large amounts of) unseen (data)?) From the findings, we conclude the following:

- Both qualitatively and quantitatively, the hybrid area-based method is superior to the point-based method. The resulting labels of the first method are more intuitive and explicable (e.g. we see clusters with homogeneous congestion; isolated wide moving jams and high complex mixed patterns); and the separability of the resulting feature space is excellent over the entire range of number of clusters tested.
- More specifically:
 - The area-based method results in (by design) directly interpretable features (e.g. disturbances, heavily congested regions, total spatio-temporal extent of congestion)
 - The obtained segmentation provides a way to process different elements in the corresponding pattern separately and allows various features to be derived. This may be beneficial in wider applications, e.g., in the identification of wave speeds or shock fronts from the segmented regions.
 - The area-based method yields a much smaller feature space and is thus (much) more parsimonious than the point based method.

However, our results do not suggest one should dismiss the point-based approach altogether. In contrast, our result highlight its capability for recognizing more detailed and subtle differences between clusters (labels). This suggests a hierarchical approach to clustering data sets may be possible:

- The area-based approach may be used as a top level approach to partition the data set based on domain knowledge.
- Subsequently, the point-based approach could be employed to further discriminate between patterns in more detail, without a priori conceptions about what these differences (physically) entail. By doing this, traffic patterns can be annotated with meaningful labels at both abstract and detail levels which potentially describe them better.

These latter bullets indicate one important line of further research (hierarchical and mixed clustering, labelling and classification). A second line of further research lies in the embedding of these methods in an overall (database) search engine that uses the reduced feature space to query the (fast growing) 200TB traffic database of the Dutch National Data-warehouse and for fast retrieval, visualisation and analysis of traffic patterns. We finally highlight some important limitations in this work. In this study, only a limited number of features are used in area-based approach; there are still opportunities for further improvement in this direction. Adding (extracting) more traffic-related features can be used for exploring typical patterns from different perspectives. Furthermore, the ability to recognize new traffic patterns is also an important topic for future research. In this respect, we seek a classifier that learns new patterns, without "forgetting" existing patterns.

Chapter 5

Feature Extraction (ii) - Bottleneck Detection and Characteristics

Highway bottlenecks are responsible for the majority of traffic congestion. Although the problem of bottleneck detection is not new, contemporary methods have not solved the problem thoroughly with regards to bottleneck locations, activation time, and related congestion tracking. These elements are essential for identifying and characterizing a bottleneck. In this chapter, we propose a comprehensive framework for detecting and extracting these features of highway bottlenecks from traffic data. We particularly focus on questions (i) whether a bottleneck is the primary source of congestion or (ii) whether it is activated due to congestion caused by another downstream bottleneck.

This chapter is an edited version of the following published paper:

Nguyen, T. T., S. C. Calvert, H. L. Vu, H. Van Lint (2021) An automatic detection framework for multiple highway bottleneck activations, IEEE Transactions on Intelligent Transportation Systems.

5.1 Introduction

Highway bottlenecks are activated when traffic demand exceeds capacity. For example, a high inflow from an on-ramp can increase the demand on the downstream road which activates a bottleneck; or the closing of a lane reduces the road capacity which might not meet its current demand and trigger a bottleneck. These bottlenecks account for the majority of congestion that occurs on highways (Hale et al., 2016; Systematics, 2004). Detecting and/or understanding the characteristics of bottlenecks, such as location, duration or delay, play a vital role in the management and control of mobility. Sensing devices like inductive loop detectors have been implemented widely to provide an essential source of information for studying traffic flow in general and bottlenecks in particular. Knowing existing bottleneck locations and their effects on traffic enables traffic experts to act quickly, albeit manually exploring or searching such data would demand considerable time and effort. To effectively utilise increasing amounts of collected data, the development of a new method that automatically analyses traffic data for bottlenecks information is necessary due to three reasons. First, by exploring traffic data automatically, such a method can simply save a lot of time and effort for network operators. Second, the automation property enables the study of bottlenecks for long periods, e.g. months or years; hence, their long-term related statistics can be obtained for further characterizing and understanding traffic bottlenecks. Finally, the method can be applied widely to large-scale freeway networks. In particular, bottlenecks on region-wide or nation-wide highway networks can be extracted automatically to study traffic network performances.

Different approaches have been proposed in the literature to identify and extract highway bottleneck characteristics automatically. Early approaches focus on pre-identified bottleneck locations, which can be learned from either network topology or historical observations. Accordingly, traffic information such as speeds or flows are obtained from related detectors (upstream and/or downstream). One can calculate the changes of speed or flow over time, and define appropriate thresholds based on historical statistics to detect the onsets and dissolves of the corresponding bottlenecks (Zhang & Levinson, 2010; Banks, 1991; Hall & Agyemang-Duah, 1991). Recent research focuses on both the spatial and temporal evolution of activated bottlenecks. Instead of individual bottleneck locations, data collected from long road segments are processed for information about multiple existing bottlenecks. Speeds or flows are normally presented by spatiotemporal maps, which are essentially matrices. Chen & Rakha (2017) proposed a set of image-processing techniques to classify traffic states into congested or non-congested. Thereafter, additional information is incorporated on the related network topology to eliminate discharging areas from congestion, and inherently identify bottleneck locations in the original congestion. However, there are challenges from the aforementioned methods that still need to be addressed. First, contemporary approaches in the literature do not distinguish between stationary bottlenecks (at fixed locations) and temporary bottlenecks that arise when so-called wide moving jams propagate upstream. This misrecognition could result in false alarms of bottlenecks.

Second, most (if not all) of the existing methods have been verified on rather simple corridors where existing active bottlenecks are away from each other, which means that they cause different (isolated) regions of congestion. Therefore, the problem of bottleneck identification is simplified to the detection of congestion. A gap remains for a method that can detect bottlenecks in more complex road corridors where multiple bottlenecks might be activated simultaneously and congestion from one of those propagates to other upstream bottlenecks. This is a relevant problem since such a method can provide more complete information about all potential bottlenecks on a corridor (or even a network), and dependencies between them can be investigated to benefit traffic management and control.

This chapter aims to develop a comprehensive framework for automatically detecting bottleneck activations in complex highway corridors. To do this, we first need to detect if there is a congestion pattern, which implies detecting its spatiotemporal extent. Afterwards, we figure out which bottlenecks contribute to the cause of this congestion. As a result, we develop a methodology (as described in Section 5.3) that contains two main parts, namely a congestion pattern detection method and a bottleneck detection method which are described in detail in Section 5.4 and Section 5.5 respectively. Our method relies solely on the discontinuities of traffic speeds over a certain time duration at (the small vicinity of) a location. Therefore, any type of bottlenecks, either due to road topologies or incidents, that induces congestion with decreasing traffic speeds at the upstream of the bottleneck locations is detected. We verify this framework with simulated data in Section 5.6 and apply it to a real case study in Section 5.7. Besides, a relevant literature review is presented in the next section.

5.2 Literature review

In this section, we review related research regarding the three main objectives, which are mentioned previously: (1) detecting the activation of a bottleneck, (2) identifying bottleneck locations and (3) tracking congestion forming upstream of a bottleneck due to its activation.

Early studies aim to identify the activation and deactivation of a specific bottleneck, of which the location is known a priori. Traffic data from e.g. inductive loop detectors are collected to provide information into traffic at the bottleneck. These data are time series showing the evolution of traffic variables like speed or flow. Banks (1990) visually inspects time series of traffic speeds on both individual lanes and aggregated over lanes of the road segment associated with a bottleneck. The drops in speeds are used as the indicator of a bottleneck activation at that location. The method is formulated by defining a speed threshold (which is derived based on experiments) to filter 30-second-interval speed differentials (Banks, 1991). Following Bank's approach, Hall & Agyemang-Duah (1991) derive a threshold of occupancy-to-flow ratio to identify the formation and dissolution of a queue. Zhang & Levinson (2004, 2010)

experimentally derive two thresholds of occupancy to classify traffic into 3 conditions: congested, uncongested, and intermediate. Then, a bottleneck activation is detected under the condition that upstream is congested and downstream is uncongested for at least a minimal time, e.g. 5 minutes. Das & Levinson (2004) incorporate and analyse both speed and flow information. In their method, traffic states are categorised into four phases: free, synchronised, congested and recovered. A decision diagram is introduced (based on various speed-flow conditions) to illustrate the changes of traffic states. Drops of speeds and flows are the fundamental metrics in this diagram, based on which bottleneck activation is identified accordingly. The authors also take into account upstream and downstream flows to argue if the onset of congestion at the current location is prone to the activation of bottleneck downstream. The aforementioned methods require two conditions that make them unsuitable for a comprehensive bottleneck detection method. First, bottleneck locations have to be known in advance; hence it can not locate bottlenecks but to detect activation of known bottlenecks. Second, they require parameters (thresholds) that are derived from manually analysing (local) related traffic data.

The second group of research incorporates both the temporal and spatial dimensions to identify the location and activation time of bottlenecks observed in data. The most popular method in this direction was introduced by Chen et al. (2004), so-called Chen's method. It processes all adjacent pairs of detectors and defines a set of rules to detect if a bottleneck is activated between two locations. These rules consist of low speeds at upstream locations, higher speeds at downstream locations, and (spatially) monotonic decreases of speeds to a certain level. Upon detecting locations and activation time of all bottlenecks on a highway, a speed threshold, which is learned through analysis of traffic data, is chosen to determine if traffic is congested. Some important characteristics can be subsequently derived such as activation frequency of bottlenecks or average traffic delay. This method might work well in processing recurrent bottlenecks and extracting characteristics for future activations, which most likely require the same parameters for the detecting method. In line with Chen's method, Zhang et al. (2018) attempted to formulate the approach in a systematic way. Specifically, the speed threshold is chosen as critical speeds, which are at the boundary between free and congested traffic, on related links. Also, a post-processing step was proposed to associate relevant activated points (indicating speed differences) together to form lines representing location and time of bottleneck activations. Bai et al. (2011) proposed a similar approach to Chen's method, though they base their bottleneck activation definition on occupancy instead of speed. As acknowledged by the authors in the original paper (Chen et al., 2004), setting parameters for these methods require visually observing historical data, and different bottlenecks will require different sets of those. For example, Wieczorek et al. (2010) conducted a sensitivity analysis of three parameters (in Chen's method) by testing 125 distinct sets to find the best combination. Although Chen's method and other approaches can be effective, their parameters are sensitive to local bottlenecks; hence, extensively applying the method to different bottlenecks will not be efficient. Recently, Yang et al. (2020) have investigated the problem from a statistical approach.

The authors proposed an optimization algorithm to estimate critical speeds by fitting the fundamental diagram (flow-speed plot) with multi-source traffic data. These critical speeds are used to detect when traffic is oversaturated. Then, the frequencies of congested states on various links can be calculated over a long period (e.g. three months in their case study). Frequent or recurrent bottlenecks are then identified by setting up a threshold on such frequencies. Hence, this approach could be suitable for identifying significantly recurrent bottlenecks instead of specific bottlenecks on a single pattern of congestion.

Activated bottlenecks cause upstream congestion, which results in slow traffic and increases travel time. Hence, to quantify the impact of a bottleneck, it is important to track the upstream congestion induced by its activation. For this purpose, traffic information like speeds is evaluated in both spatial and temporal dimensions, which constitute a 2D numerical matrix. Different methods have been proposed in the literature. As a follow-up improvement of Chen's method (Chen et al., 2004), Bertini & Horowitz (2008) added two more rules to mark the upstream congestion on the related spatio-temporal map of traffic state if and only if (1) downstream detector is labelled as congested, and (2) the speed at the current detector is below a speed threshold. The latter condition assures that only congested (space-time) points associated with bottlenecks are identified, which means low speeds due to disturbances and not caused by bottlenecks are ignored. Palmer et al. (2009) suggest combining Chen's method with the FOTO model (Forecasting of Traffic Objects) and the ASDA model (Automatic tracking of moving traffic jams) introduced by Kerner et al. (2004) to improve the reliability of the resulted bottleneck detecting system. The emergence of different patterns of upstream congestion related to bottlenecks identified by Chen's method can be detected and their evolutions can be tracked by Kerner's methods. Analysing them can yield further details such as wave speed and travel time loss, which are relevant information of bottlenecks. Another direction of research classifies traffic states into two common states, namely congested and free flow. Ban et al. (2007) use percentile speeds on multiple days of traffic data to identify regularly recurrent positions of bottlenecks. Then a speed threshold is chosen to binarize the (percentile-) speed map. Jin et al. (2012) proposed coordinate transforming of the flow-density diagram into a different feature space (so-called Uncongested Regime Shift (URS) - Perpendicular to Uncongested regime Shift (PUS)) and used PUS as the indicator for congested traffic state. A threshold is determined experimentally. As a result, a congestion contour map of a corridor is obtained by calculating congestion frequencies at all detectors over time. These methods give a statistical view of which road stretch and how often it is affected by bottlenecks rather than the specific location and activation time of a bottleneck. Elhenawy et al. (2013) proposed a bottleneck identification algorithm based on the assumption that traffic states exist in two different phases, i.e., congested and free flow, and speeds in these phases follow two Gaussian distributions. Accordingly, a t-test is conducted on each space-time point to classify traffic state into one of these phases. The distribution of speeds can vary significantly between different highways; hence, their parameters need to be well adjusted before being further applied. Using

the same assumption, Chen & Rakha (2017) generalize the problem of classifying traffic states into two categories to image binarization, which is a well-developed topic in computer vision. The authors proposed using Otsu's method which, in essence, minimizes speed variances in individual categories as well as maximizing their variances across categories. In addition, road geometry is incorporated to separate bottlenecks which might be connected in a speed map, i.e. congested traffic from one bottleneck propagating upstream and merging to congested traffic of the downstream one. One of the main drawbacks of this approach is that the location of a bottleneck is an uncertain quantity. For example, the activation can occur at any place downstream of an on-ramp. Hence, although combining different sources of traffic data is a reasonable approach, further research is required to enable extending this method to different locations or roads.

Activations of bottleneck dampen vehicular speeds and the effect is visually strongly observable in speed maps. In addition, applications of traffic state estimators can result in satisfactory views of traffic over both the spatial and temporal dimensions with equidistant resolutions. The Adaptive Smoothing Method (Treiber & Helbing, 2002; Schreiter et al., 2010a) is a simple filter yet significantly effective estimator for filling in traffic information (e.g., speeds) between detector locations. The resulted speed maps are, therefore, easily seen/treated as images. Besides, computer vision is a well-developed field where tools and techniques are available for wide-range applications. Hence, image processing techniques have naturally come as a suitable approach for analysing bottleneck activations.

In summary, the literature offers a range of methods to classify traffic patterns and determine bottleneck locations and properties. However, these methods may not work on road corridors with several bottlenecks in close vicinity. In this chapter, we propose a comprehensive method from an image-processing approach to (i) automatically detect locations and activation/deactivation time of highway bottlenecks, and (ii) track the congestion resulted upstream. Importantly, the method is able to disentangle different bottlenecks in complicated congestion existences in which multiple bottlenecks are being activated concurrently and causing joint platoons of traffic jam. The related upstream congestion is identified to quantify the impacts of these bottlenecks on traffic. The method can be easily extended to different highways to efficiently assist (large-scale) studies of traffic bottlenecks.

5.3 Proposed framework

This section presents our proposed framework for the problem of bottleneck detection and associated congestion identification, as illustrated in Fig. 5.1. Overall, the bottleneck detection method locates potential bottlenecks from traffic speeds. The results are validated using other relevant sources of data, for example, road geometry can be used to evaluate detected locations whether they are reasonable. The detection method,

which is the core of our framework, consists of four main modules, namely congestion detection, speed discontinuities detection, activation location and time identification, and refinement.

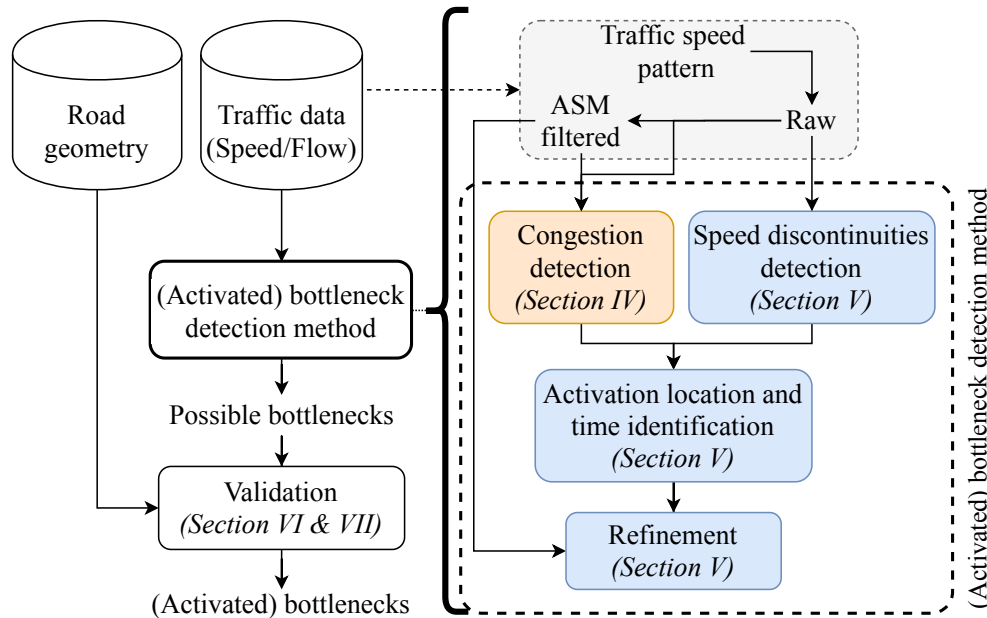


Figure 5.1: The overall framework of the proposed methods for bottleneck detection

The first module classifies speed measurements into either congested or uncongested states. Given a speed map representing traffic on a route over time, this module essentially identifies congested regions which are our main region of interest. Additionally, this also plays an important role in associating relevant (congested) regions to different bottlenecks that are going to be detected. Hence, together they can provide a more complete picture of bottlenecks including activations and consequences. The second module aims to highlight speed discontinuities in the traffic pattern since these are probably the first necessary (and easily observable) evidence for the existence of bottlenecks. This module also takes into account wide moving jams that introduce speed reductions (although they are not necessarily static bottlenecks). The embedded method can reduce their effects on detecting bottleneck-related discontinuities of traffic speeds. Based on the highlighted speed disruptions, the next module identifies potential bottleneck activations therein. To do so, it needs to gather and cluster highlighted points by incorporating their spatial and temporal information. The goal is to separate points associated with different activations of bottlenecks. Notice that, the previous modules process data from loop-detectors (so-called raw data); hence, the outcomes, i.e. activated locations, are locations of detectors.

In the proposed framework, raw data and ASM data are used in different modules to utilize their advantages. Raw data are direct measurements collected from loop detectors and ASM data are obtained from applying the Adaptive Smoothing Method on the raw data. While we only have traffic measurements at sparse locations where

loop detectors are available, ASM further estimates traffic data at equidistant locations and provides a more complete view of traffic therein. In our framework, ASM data are used for detecting congestion because an image-based representation of a traffic pattern is more efficiently constructed from ASM data as compared to raw data. This inherently benefits the identification of the originality of a bottleneck activation, i.e. whether is is a primary or secondary bottleneck.

5.4 Congestion detection

This section presents a new approach based on image processing methods for congestion detection in the first module. Given an image representation of a congestion pattern, the objective is to detect various regions that are associated with congested traffic. We propose to formulate this as an image segmentation problem in which the target is to discern foreground objects from background areas which represent congestion and uncongested traffic respectively. We first introduce a well-known approach which is the so-called Chan-Vese model (Chan & Vese, 2001) from a general view on object tracking. Afterwards, we show how the model is used to formulate the congestion detection problem.

5.4.1 The Chan-Vese model

The Chan-Vese model (Chan & Vese, 2001) is an active contour model which evolves a curve to boundaries of objects in images. The main principle of the algorithm is to minimize an energy function $F(c_1, c_2, C)$ defined as:

$$\begin{aligned}
 F(c_1, c_2, C) = & \mu \text{Length}(C) + \nu \text{Area}(\text{inside}(C)) \\
 & + \lambda_1 \int_{\text{inside}(C)} |u(x, y) - c_1|^2 dx dy \\
 & + \lambda_2 \int_{\text{outside}(C)} |u(x, y) - c_2|^2 dx dy
 \end{aligned} \tag{5.1}$$

where $u(x, y)$ is a given intensity image, C is any variable curve, c_1, c_2 are average intensity values of u inside and outside C respectively, $\mu \geq 0, \nu \geq 0, \lambda_1, \lambda_2 > 0$ are fixed parameters. The solution C is at the boundaries of foreground objects in the image. For details of explanation or justification, we refer the readers to the original paper (Chan & Vese, 2001).

The minimization problem can be solved by using the level set method (Osher & Sethian, 1988) which describes all computations on a level set function ϕ having the following features

$$\begin{cases} \phi(x, y) = 0, \text{ for } (x, y) \in C \\ \phi(x, y) > 0, \text{ for } (x, y) \in \text{inside}(C) \\ \phi(x, y) < 0, \text{ for } (x, y) \in \text{outside}(C) \end{cases} \tag{5.2}$$

The energy function is updated as a function of ϕ (see Equation 5.3) instead of C .

$$\begin{aligned}
F(c_1, c_2, \phi) &= \mu \int \delta(\phi(x, y)) |\nabla \phi(x, y)| dx dy \\
&+ \nu \int H(\phi(x, y)) dx dy \\
&+ \lambda_1 \int |u(x, y) - c_1|^2 H(\phi(x, y)) dx dy \\
&+ \lambda_2 \int |u(x, y) - c_2|^2 (1 - H(\phi(x, y))) dx dy
\end{aligned} \tag{5.3}$$

where, H is the Heaviside step function and δ is the delta function; their definitions are shown in Equation 5.4.

$$\begin{aligned}
H(z) &= \begin{cases} 1, & \text{if } z \geq 0 \\ 0, & \text{if } z < 0 \end{cases} \\
\delta(z) &= \frac{dH(z)}{dz}
\end{aligned} \tag{5.4}$$

Consequently, a curve C can be defined implicitly by the zero-level set of the function ϕ (i.e. set of points with $\phi(x, y) = 0$). Accordingly, the motion of a curve can be represented efficiently and easily by tracking the zero level set of the function ϕ . The minimization of $F(c_1, c_2, \phi)$ can be solved by constructing the Euler-Lagrange equation for ϕ (noting that c_1, c_2 are dependent and easily calculated from ϕ). To satisfy the differential condition, a small adjustment is made to make the Heaviside step function and the delta function differentiable at around location $z = 0$. We call these adjusted versions H_ε and δ_ε ; as $\varepsilon \rightarrow 0$ they converge to H and δ respectively. Now, the evolution of ϕ (over virtual time t) is described by the following Euler-Lagrange equation

$$\frac{\partial \phi}{\partial t} = \delta_\varepsilon \left[\mu \nabla \cdot \frac{\nabla \phi}{|\nabla \phi|} - \nu - \lambda_1 (u - c_1)^2 + \lambda_2 (u - c_2)^2 \right] \tag{5.5}$$

From Equation 5.5, the evolution of ϕ is controlled by two terms: the curvature $\kappa = \nabla \cdot \frac{\nabla \phi}{|\nabla \phi|}$, which preserves its smoothness and the so-called region term $-\lambda_1 (u - c_1)^2 + \lambda_2 (u - c_2)^2$ affects the motion of the (zero-level set) curve.

5.4.2 The Chan-Vese model for traffic congestion detection

In traffic congestion detection, we aim to detect the curve C that surrounds congestion regions presented in spatio-temporal speeds of a given pattern. This speed pattern is equivalent to an intensity image $u(x, y)$ where pixel values represent traffic speeds at corresponding locations x and time y . The congestion region is the *inside*(C) and the free flow traffic region is the *outside*(C). Based on this notation, our congestion detection problem can be solved by applying the Chan-Vese method to the equivalent image of traffic speeds. In Section 5.6 we will elaborate the Chan-Vese method step-by-step and illustrate these steps with an example traffic data set (e.g. Fig. 5.3 and Fig. 5.4). Before doing so, we first explain the second key component of our methodology, which is the bottleneck identification method.

5.5 Bottleneck identification

In this chapter, we aim to detect activations of bottlenecks in two situations. Specifically we are interested in (i) whether a bottleneck is the primary source of congestion or (ii) whether it is activated due to congestion caused by another downstream bottleneck. We refer to these situations as primary and secondary bottlenecks. In the activation of the former, there is no congestion downstream of the corresponding bottleneck, meaning traffic is moving freely; whereas in the latter case, downstream of the bottleneck is congested due to another bottleneck (further downstream). In a dense network where there are multiple (topologically potential) bottlenecks located close to each other, congestion due to an activation of a bottleneck can propagate upstream and trigger other bottlenecks. Disturbances can emerge and possibly turn into wide moving jams which can pass through upstream bottlenecks. These factors might hinder the detection of activation of these secondary bottlenecks for (at least) two reasons: (1) interruptions of traffic speeds at secondary bottlenecks are normally less significant as compared to those at primary bottlenecks since traffic speeds are already low when approaching these locations, and (2) the speed changes are interfered with wide moving jam which can be observed more clearly along the direction of those jams. To avoid (falsely) recognizing the former phenomenon with any other speed disruptions (which are not due to bottlenecks), one would need to observe the disruption on a temporal dimension to test for longevity. Only if traffic has been congested for a certain long period, a bottleneck can be a possible reason (though another possibility is incidents). Regarding the second reason, it is generally accepted (i.e. there is abundant evidence (Cassidy & Windover, 1995; Kerner & Rehborn, 1996; Treiber & Helbing, 2002; Schreiter et al., 2010b)) that the dominant congestion wave speed is in the vicinity of -20km/h (the negative sign indicates opposite direction of traffic); hence, by introducing a filter along this direction, one can expect to eliminate the interference of wide moving jam in the detection of activations of secondary bottlenecks.

Based on the above observations, we have identified and developed a method for detecting and identifying both location and activation time of bottlenecks, especially in dense networks where there are multiple bottlenecks in close vicinity.

5.5.1 Speed discontinuities detection

In the spatio-temporal representation of traffic, a bottleneck activation is observed by (temporally lasting) decreases of vehicular speeds at a certain location (or a vicinity thereof). This phenomena holds in bottlenecks caused by either road topologies or incidents. To identify bottleneck activation, we first detect speed discontinuities along the direction of traffic flow under congested condition. In congested traffic disturbances propagate against the direction of traffic flow. Accordingly, gradients are calculated in this direction to highlight the disruptions (if they exist) of traffic speeds.

Below we develop a method to construct and apply an appropriate gradient kernel for that purpose.

Given a traffic speed pattern represented by intensity image u , Equation 5.6 shows the procedure of calculating gradients, G^c , along congested waves. The kernel P_c is defined by rotating a longitudinal gradient kernel P , which calculates speed differentiations on the spatial dimension. The size of the kernel determines how many related neighbouring pixels contribute to the speed discontinuity of a central pixel. Throughout the chapter, traffic speed map u is presented in a way that the driving direction is from bottom to top, and that is the decreasing order of indices in u .

$$\begin{aligned}
 G^c &= u_0 * P_c \\
 P_c &= \text{rotate}(P, w_c) \\
 P &= \begin{bmatrix} +1 & +1 & +1 \\ -1 & -1 & -1 \end{bmatrix} \\
 w_c &= \text{wave speed} \approx -20\text{km/h}
 \end{aligned} \tag{5.6}$$

One way to approximate the kernel P_c is to propagate the top row of the Prewitt kernel P to the bottom row with the speed of w_c , assuming the distance between them is Δy . This is to mimic the propagation of traffic waves in congestion. Its translated position is calculated and the corresponding values in P_c are determined by discretization afterwards. We propose a procedure as follows (see Fig. 5.2 for an illustration).

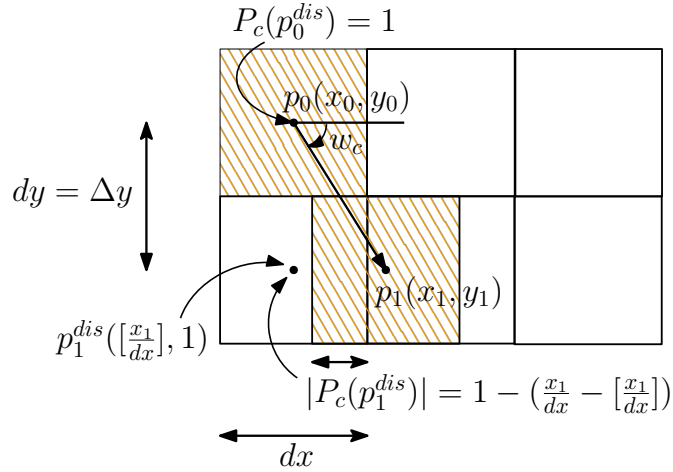


Figure 5.2: A method to approximate the kernel P_c . dx, dy are temporal and spatial resolution, respectively.

- (i) Define a coordinate system to P with left-right and top-down as positive directions. Pick the top-left pixel and assign its coordinate as $p_0 = (x_0, y_0)$.
- (ii) Translate p_0 downward with speed w_c and obtain $p_1 = (x_1, y_1)$. p_1 is identified based on the following equations. (Δy is the distance between two consecutive

locations.)

$$\begin{aligned} y_1 &= y_0 + \Delta y \\ x_1 &= x_0 + \frac{y_1 - y_0}{w_c} = x_0 + \frac{\Delta y}{w_c} \end{aligned} \quad (5.7)$$

- (iii) Now comes the discretization step with the spatial and temporal resolution (dx, dy) . By assuming the distance between two rows is the unit distance, we get $dy = \Delta y$. We determine which (leftmost) item that p_1 sits on and its P_c value accordingly. This can easily be done by discretizing x_1 by the (temporal) unit dx . Accordingly, its P_c value is proportional to the intersection of cell p_1 and its discretized cell p_1^{dis} .

$$\begin{aligned} p_0^{dis} &= (0, 0), & p_1^{dis} &= \left(\left[\frac{x_1}{dx}\right], 1\right) \\ P_c(p_0^{dis}) &= 1, & |P_c(p_1^{dis})| &= 1 - \left(\frac{x_1}{dx} - \left[\frac{x_1}{dx}\right]\right) \end{aligned} \quad (5.8)$$

- (iv) Fill all items on the right of p_1^{dis} with 1's and those on the left with 0's. Then construct a symmetric P_c with respect to its central item. Finally, change the sign of all values in the bottom row to negative.

This procedure can be expanded to determine kernels with more elements if needed. Note that, the above procedure uses the direct (mathematical) gradient kernel as the underlying kernel, one might as well use different kernels such as Prewitt or Sobel.

5.5.2 Activation location and time identification

If a bottleneck is activated for a period, one can observe a speed disruption during that time. Alternatively, the response (G_c) of the P_c -based filter presents minimal (negative) values at related time and locations. Due to various reasons e.g. heterogeneity of traffic or disturbances of traffic at bottleneck location or noises in measurements, these negative values are not only found at the bottleneck location but also nearby locations. Besides, these locations also spread horizontally as long as the related bottleneck is activated. To aim for a robust method, we group pixels with negative values (in the response G_c) into rectangular clusters which each of them represents the speed disruption of a potential bottleneck spatially and temporally. We refer to them as *bold lines*.

A mathematical approach for this clustering problem is to determine rectangular boundaries that minimize intra-distances of pixels inside the same boundaries, pixels outside boundaries and/or maximize intra-distances between pixels inside boundaries and pixels outside boundaries. Despite that, in this section, we propose an algorithm to solve the problem with a simplified yet effective approach. The underlying principle is to find all minimum and extend the corresponding boundaries until they reach edges with

average values approximately the same as the background value. The main steps of the algorithm are as follows (see Algorithm 5.1 for a summary):

- (i) Estimate the background response value by averaging negative responses outside the congestion area.
- (ii) Identify local minima in F by comparing each value with all eight of its neighbours. We denote this set as \mathcal{M} .
- (iii) Pick the smallest minimum in \mathcal{M} . For each side in {left, right, top, bottom}, calculate average G^c values. If it is larger than the estimated background value G_{bkg}^c , expand the boundary to include this edge. Iterate this procedure until no more expansion is possible. As a result, a (rectangular) boundary of the region surrounding some minimal G^c can be determined. Next, remove all the minima in this region from \mathcal{M} and iterate the process until \mathcal{M} is empty.
- (iv) Bottleneck locations and activation time: For each one of the rectangular regions found in the previous step, the location and activation time of the related (potential) bottleneck is identified by finding the line with the strongest sum/average of G^c values. A map of all these lines, indicating all possible bottlenecks in the given pattern, is obtained.
- (v) Refinement: Relevant rules are applied to clean unnecessary lines, for example very short lines due to disturbances or noise. One can define the minimum activation time for bottlenecks of interest and eliminate lines with shorter lengths accordingly. Also, lines or their parts that lie outside of the congestion mask, identified in the previous section, are eliminated.

5.5.3 Identification of associated congestion

The previous sections have shown how to (1) detect congestion regions in a spatio-temporal speed map, and (2) identify lines which indicate locations and activation time potential bottlenecks therein. Based on these two elements, congestion regions associated with detected bottlenecks can be identified. There are two underlying principles in this algorithm. First, a spatio-temporal congestion region is attached to the most downstream bottleneck (if it exists). Second, when a bottleneck is triggered, the activation continues until congestion dissolves.

5.5.4 Primary or secondary bottleneck determination

In this section, we propose an algorithm to determine if a bottleneck is primary or secondary. It is based on the congested regions associated with the detected bottlenecks.

Algorithm 5.1 Identification of bottleneck activation location and time

Require:

- Response G_c of a speed pattern to kernel P_c
 Congestion (image) region indicator, or congestion mask, M_c
-

I - Background value estimation

- 1: Free flow mask $M_f = \overline{M_c}$
 2: Background filter response $G_{bkg}^c = \frac{\sum_{p \in M_f} G^c(p)}{|M_f|}$
-

II - Local minimum

- 3: $\mathcal{M} = \{m | m \in G^c, m \text{ is a local minima}\}$
-

III - Bold lines identification

- 4: **while** $\mathcal{M} \neq \emptyset$ **do**
 5: $m_i \leftarrow \underset{m \in \mathcal{M}}{\operatorname{argmin}} G^c(m)$
 6: $r_e \leftarrow$ the rectangular boundary of m_i
 7: **while** r_e is expanding **do**
 8: **for each** neighbour edge e of r_e **do**
 9: **if** $\frac{\sum_{p \in e} G^c(p)}{|e|} \geq G_{bkg}^c$ **then**
 10: $r_e \leftarrow r_e \cup e$
 11: **end if**
 12: **end for**
 13: **end while**
 14: $\mathcal{R} \leftarrow \mathcal{R} \cup r_e$
 15: $\mathcal{M} \leftarrow \mathcal{M} \setminus r_e$
 16: **end while**
-

IV - Location and time identification

- 17: **for each** $r_e \in \mathcal{R}$ **do**
 18: $s \leftarrow$ vertical projection of $G^c(r_e)$
 19: Activation location $l \leftarrow \underset{x \in s}{\operatorname{argmin}} s(x)$
 20: Activation time t as in r_e
 21: **end for**
-

V - Refinement

- 22: Apply relevant constraints to eliminate unrelated lines
-

By identifying traffic states downstream of a bottleneck, i.e. their related congested region, the source of the corresponding congestion can be identified effectively. Particularly, the condition for a primary bottleneck is that its downstream traffic is not congested, meanwhile, congestion has already occurred downstream during the begin-

ning of a secondary bottleneck. Our proposed procedure for identifying primary or secondary bottlenecks is shown in Algorithm 5.2.

Algorithm 5.2 Primary or secondary bottlenecks classification

Require:

\mathcal{C} : includes a separation of related congestion regions of detected bottleneck activations

Procedure

- 1: **for each** $b^i \in \mathcal{B}$ **do**
 - 2: t_0^i is when b^i is triggered/activated (which is associated with the top-left pixel of c^i)
 - 3: $d \leftarrow$ downstream regions of c^i at time t_0^i
 - 4: **if** d is not congested **then**
 - 5: b^i is a primary bottleneck
 - 6: **else**
 - 7: b^i is a secondary bottleneck
 - 8: **end if**
 - 9: **end for**
-

5.6 Methodology verification

In this section, we verify the two main components of the proposed method, namely traffic congestion detection and bottleneck identification. The former is compared with the bimodal-based method, a well-known method for the classification of traffic into either congested or uncongested states. For the latter, simulated data is used due to their advantages over real data.

5.6.1 Traffic congestion detection

To evaluate the performance of the proposed approach on classifying traffic states in congestion patterns, we first analyse the parameters in the Chan-Vese model. Then, we compare our approach with the bimodal-based method (Chen & Rakha, 2017), which is the most popular one found in the literature.

For the Chan-Vese model to converge quickly and precisely at the boundaries of congestion regions, it is necessary to initialise the zero-level set curve, ϕ_0 , close to the congestion boundaries. We have tested different initializations of ϕ_0 with various values in this range and have come to the same (expected) conclusion. Namely, using speed thresholds in between 30 and 60 km/h increases the reliability of congestion classification in traffic patterns by the Chan-Vese model. As a demonstration, Fig. 5.3 shows the final contours with respect to different initial settings of ϕ_0 . The presented

traffic goes through two road stretches with different speed limits which impose different free speeds, congestion occurs in the downstream lower speed region and slightly reaches the higher speed region. The energy function minimization (Equation 5.3) has two (local) solutions on this pattern. Different initialisations of ϕ_0 lead to different classification of the pattern. If a high speed (e.g. 75km/h) is used, ϕ converges to the function whose zero-level set is at the boundary with high free speeds upstream and low speeds downstream (see the line of ϕ_0 in Fig. 5.3b). Consequently, the deduced congested region covers (almost) the whole region with low free speed, which is not the desired result. On the other hand, by starting with low congested speeds (e.g. 40km/h), the converged ϕ is at the boundary of the congestion that we observe from the pattern (see Fig. 5.3a). Hence, by starting ϕ_0 at the speed of 40km/h, the expected congested region is identified sufficiently by the Chan-Vese model.

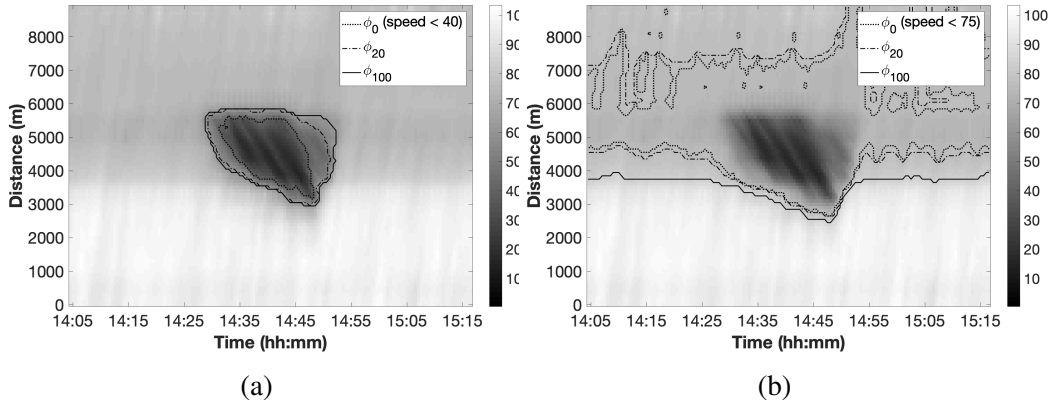


Figure 5.3: Evolution of the zero-level set of ϕ according to different initializations (a) initial mask is 40km/h (b) initial mask is 75km/h.

In addition, two different scenarios have been used to compare the method with the bimodal-based method. In the first scenario, only one free traffic speed is available in congestion patterns. Note that fluctuations of this free speed are normally observed from traffic data. The second example has at least two different free speeds in congestion patterns. Fig. 5.4 shows two examples of each of these scenarios and the corresponding outcomes of the two methods. As shown in the figure, both perform well on the two topmost patterns. A quantitative comparison indicates that their identified congested regions overlap by more than 99%. This shows that the proposed method based on the Chan-Vese model delivers comparable results as that of the bimodal-based method in simple layouts of road stretch, on which traffic speeds can be separated into two distributions. On the other hand, the bottom two patterns are two examples where the assumption of the bimodal does not hold. The presence of different road stretches with various speed limits has led to unexpected results when applying the bimodal method as shown in the middle figures. The congested regions are over-identified to include the lower free speed regions of the downstream road stretch. One might suggest to look for a local optimum of speed distributions in these patterns and identify the one that most likely represents expected congestion boundaries. Having said that,

this depends greatly on the histogram of traffic speeds and such congestion-related optimum are not clearly shown and/or easily identified. Unlike the results from the bimodal method, those from the Chan-Vese method do not cover the entire regions with lower free speed. Qualitatively, they accurately cover the congested regions in the related patterns (as can be seen from Fig. 5.4c). This has shown the superiority of the Chan-Vese method over the bimodal-based method in detecting congestion in traffic patterns.

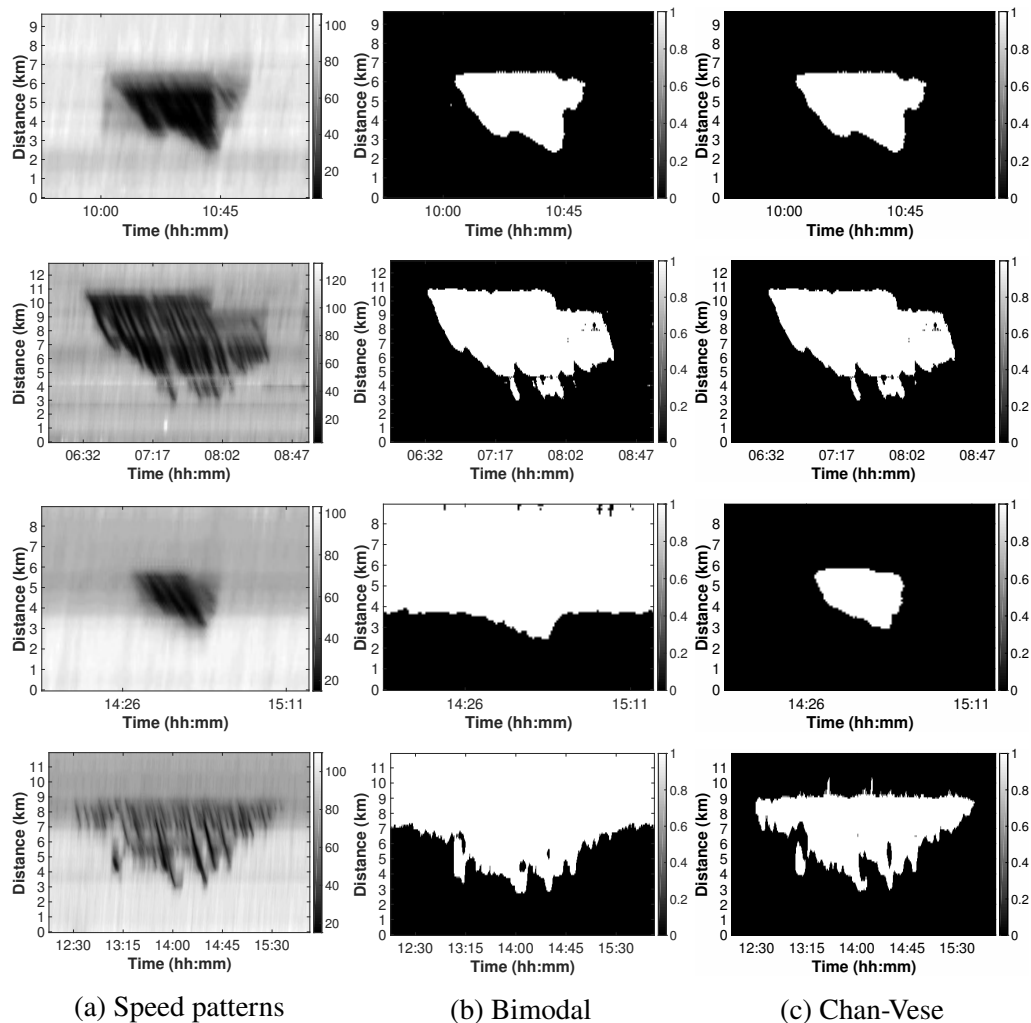


Figure 5.4: Comparison of the Chan-Vese model and the Bimodal-based method on different congestion patterns: the top two patterns have one speed limit, while the bottom two have two different speed limits. The congestion detection results of applying the bimodal method and the Chan-Vese model are shown in (b) and (c) respectively.

Through experimenting with the proposed approach, using the Chan-Vese model for the detection of congested regions in traffic patterns, it is positive to conclude that the Chan-Vese method is a highly viable method that can perform well on different congestion patterns.

5.6.2 Verification of the bottleneck identification method

Verification approach

For verification of the proposed method, we aim to analyse: (1) its capability of detecting bottleneck activations in congestion patterns, (2) how the setting of loop-detectors affects the method's outcomes. For these objectives, we make use of traffic simulation to produce crisp data that is difficult to find from real traffic flow data. In particular, a microscopic traffic simulator can provide granular details into traffic such as vehicle trajectories, traffic speeds on every short distance interval (by simply setting up loop-detectors). Note that these cannot be provided by raw traffic data due to limited numbers of loop detectors. These features enable us to identify ground-truths of bottleneck activation locations, which is necessary for evaluating the accuracy of the proposed method. Additionally, by manipulating loop-detectors (in a simulator), we can test if the method is capable of detecting activations of bottlenecks in deduced traffic patterns and analyse how those settings affect the outcomes. Following are the steps carried out.

- Design a road stretch with possible close bottlenecks that are activated concurrently with a high traffic demand. This ensures that a test can be performed with heavy congestion.
- Set up loop-detectors with short intermittent distances, i.e. 100 meters, to record traffic data. This provides a convenient base for changing the loop detector setups later on. For example, we can eliminate loop detectors to get coarser traffic patterns.
- Tuning incoming traffic flows to activate one or more of these bottlenecks.
- Repeatedly apply the proposed method and investigate the results.

Details of these steps are in the next sections.

Simulated example design In this study, we use the microscopic simulation tool FOSIM (Freeway Operations SIMulation) (Vermijs & Schuurman, 1994) which was developed at Delft University of Technology. It models traffic dynamics through the simulation of the behavior of individual vehicles. An artificial road stretch is designed as shown in Fig. 5.5. This road stretch has several potential bottlenecks which are two on-ramps, one off-ramp and a road split. While the first on-ramp (ON1) is located further, 2500 meters away, from the next (potential) bottleneck, the second on-ramp (ON2) is quite close to the off-ramp (OFF1) which is just 500 meters downstream. This is expected to create a complicated weaving section which will trigger congestion. The road split is designed to also create a bottleneck when vehicles have to change lanes to meet their desired destinations.

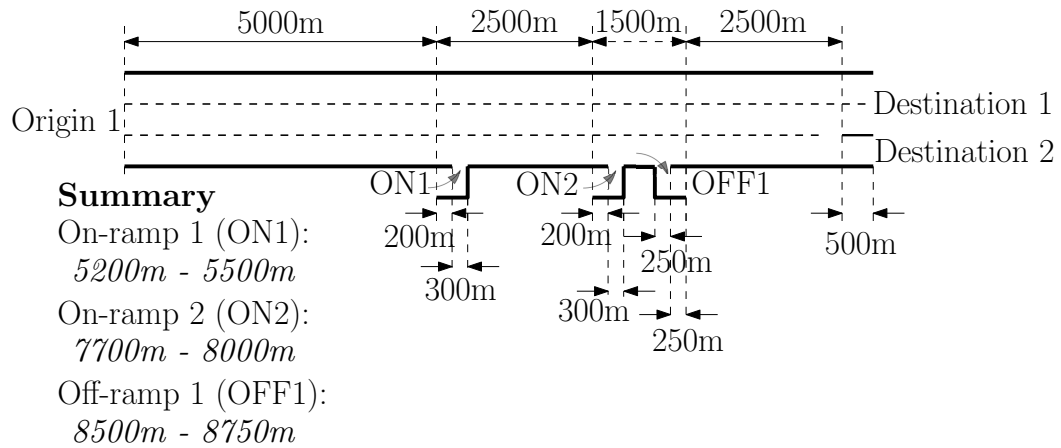


Figure 5.5: The layout of the example (simulated) road stretch.

Loop detectors are implemented every 100 meters (along this 12km road stretch) and record traffic every 60 seconds. Hence, we can obtain fine simulated traffic data for further investigation. This is much better than in reality where loop detectors can be 300 meters to more than 1000 meters apart.

Simulated congestion patterns Fig. 5.6 shows spatio-temporal speed maps of congested traffic obtained from the FOSIM model on the road layout in Fig. 5.5. As illustrated in the figure, two bottlenecks have been activated. The first one (ON2) is at a distance of around 8000-meter from the Origin 1, and the second one (ON1) is at a distance of around 5200-meter from the Origin 1 (see Fig. 5.5 for the road schematic). For simplicity, from here on we use relative distances from the Origin 1 to identify different locations on the simulated road stretch. The congestion triggered by the downstream bottleneck propagates further upstream and reaches the upstream bottleneck. Hence, we have selected this as a typical example to verify the proposed method. We have also varied distances between loop detectors to generate different levels of details in raw data. Fig. 5.6 shows three deduced patterns with spacings of 300, 500, and 1000 meters between two consecutive detectors. Two activated bottlenecks can be observed from these patterns clearly, although it is more difficult with those in the p1000 pattern (Fig. 5.6d).

Verification results

The proposed method is applied to all the simulated patterns shown in Fig. 5.6. Summary of the results is given in Table 5.1. It is used to answer two questions: (1) Is the method capable of detecting bottleneck activations, and (2) Do the locations extracted from different deduced patterns consistently point to the same locations? The former, once confirmed, will show the effectiveness of the method, while the latter will demonstrate its reliability.

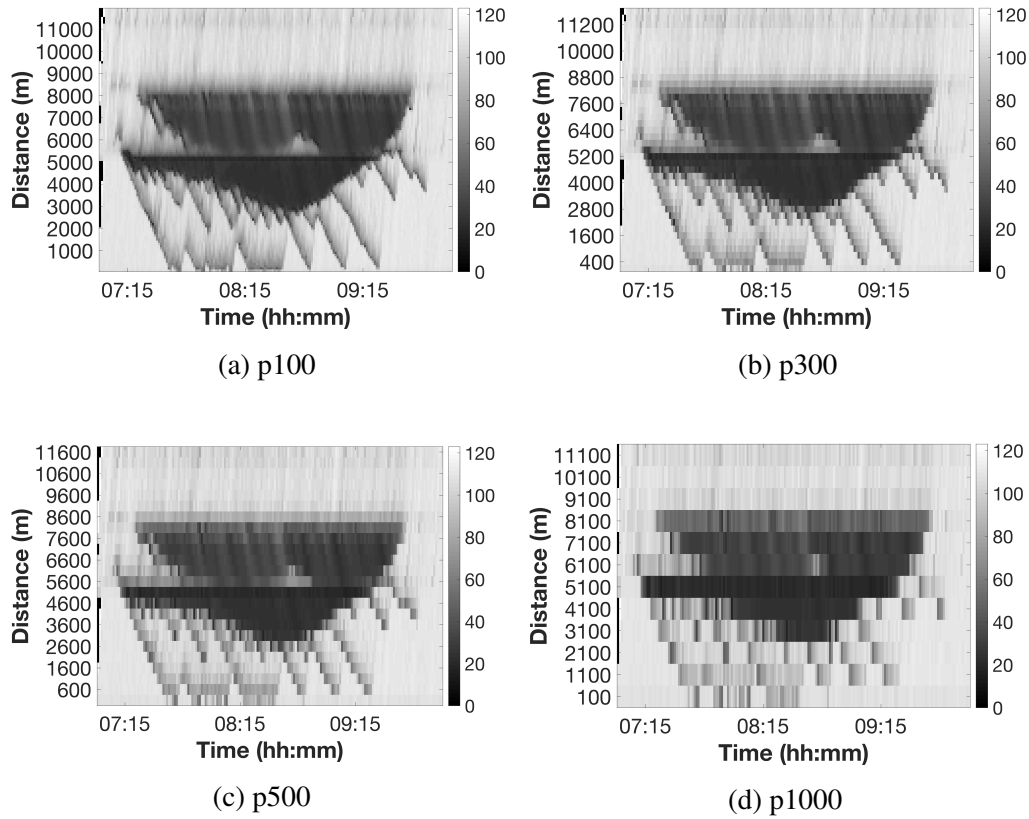


Figure 5.6: Simulated traffic patterns: (a) the original pattern with detectors at every 100 meters, and other deduced patterns which are obtained by eliminating loop detectors to maintain detector spacing distances of 300m (b), 500m (c), or 1000m (d).

The results indicate that the proposed method has successfully detected the activations of the two major bottlenecks in all the simulated patterns. Consequently, this simple experiment has shown the capability of the proposed method in detecting bottleneck activations or speed discontinuities in congestion patterns.

The results show that most of the associated detectors (of detected bottlenecks) are close to the activation points. In particular, the ON1 bottleneck is detected somewhere downstream of the 5200m or 5100m detectors, which are very close to the actual activated location - 5200m. Similarly, those detectors related to the ON2 bottleneck are located at 7900m, 8000m, 8100m which are also close to the activation point - 8000m. To correctly interpret these results, notice that raw data can only give rough estimates of activated bottlenecks, i.e. the locations of closest upstream and downstream detectors. Therefore, the actual locations might be anywhere in between. If intermediate locations (between detectors) are used as predicted activation points, the error that the method on raw data incurs grows as the detector spacing becomes larger (see the table for details). For strong bottlenecks, like the ON1, the actual locations (5200m) are in between the detected pairs of associated detectors. Whilst this is not always the case with weak bottlenecks like the ON2, for which the detected pair (8100m-8600m) does

Table 5.1: Summary of detected bottleneck locations obtained by using raw loop-detector data. These locations are the upstream and downstream detector locations with respect to detected bottlenecks.

Detector space (m)	Detected locations	Error Offset
<i>Bottleneck: On-ramp 1 (ON1) 5200m</i>		
100	5200-5300	50
300	5200-5400	100
500	5100-5600	150
1000	5100-6100	400
<i>Bottleneck: On-ramp 2 (ON2) 8000m</i>		
100	8000-8100	50
300	7900-8200	50
500	8100-8600	350
1000	8100-9100	600

not cover the actual activated location (8000m). There are two causes for explaining this. First, speed accelerations, i.e. magnitudes of speed discontinuities, change more sharply with stronger bottlenecks, therefore it is easier to detect their peaks. Second, since we calculate differences of speeds at locations of detectors, how those detectors are implemented also affects the accuracy of the detecting results. In particular, detection of stronger bottlenecks are more sensitive to this.

5.6.3 Time complexity

The method for traffic congestion detection is based on the Chan-Vese model. The numerical solution proposed in the original paper (Chan & Vese, 2001) evolves the initial zero-level set over a predefined number of iterations, η . With a limited η , this method of detecting congestion has a time complexity of $\mathcal{O}(|E| \times |T|)$, where, $|E|, |T|$ are the sizes of spatial length and temporal duration of the (ASM-based filtering) speed map, respectively.

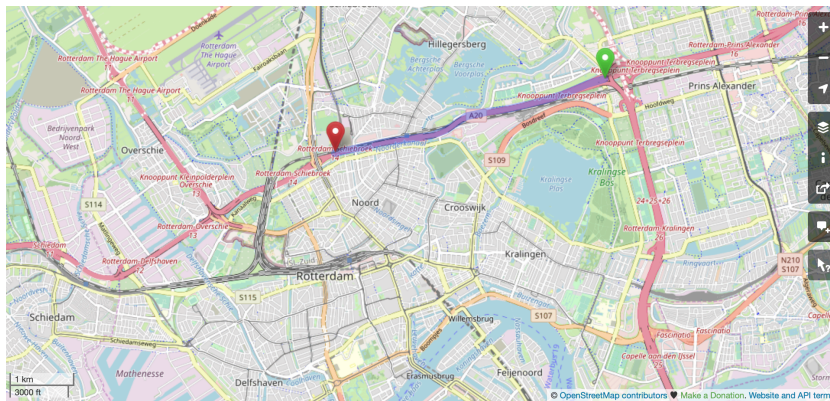
The time complexities of the three main components of the bottleneck identification method are as follows. Speed discontinuities are detected by filtering through all *pixels* in the speed map, hence, it has a complexity of $\mathcal{O}(|E| \times |T|)$. Activation location and time are identified from raw data with the complexity of $\mathcal{O}(|E^r| \times |T^r|)$, where, $|E^r|, |T^r|$ are the sizes of spatial length and temporal duration of the raw speed map, respectively. Observe that this is (greatly) dominated by $\mathcal{O}(|E| \times |T|)$ as filtered data generally has higher resolutions than raw data.

By combining all the components complexities, it is expected that the complexity of our proposed framework is linear to the size of ASM-based filtering speed map. It is

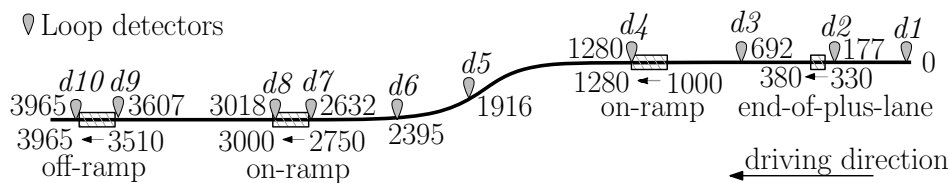
also worth to note that actual processing time also depends on the selection of parameters, e.g. the number of iterations in the Chan-Vese model.

5.7 Case study

This section demonstrates an application of the proposed method. Given a route with multiple topological disruptions, like on-ramp or off-ramp, the objective is to study (1) which are the most frequently triggered bottlenecks and (2) are they the primary source of congestion, i.e. primary or secondary bottlenecks. Also, we aim to have the answer over a long period, e.g. one year long, so that derived statistics can give more general overview on the road stretch. For that, an automatic method like the proposed one is highly relevant.



(a) The studied corridor on OpenStreetMap



(b) A simple schematic of the road

Figure 5.7: The studied corridor is on the ring road of Rotterdam, the Netherlands. The relative distances of the detectors are shown next to the detector symbols. Estimated distances of road topologies are shown in pairs of (begin, end) distances.

We have selected a corridor on the ring of Rotterdam, the Netherlands, to study (see Fig. 5.7a for a snapshot from OpenStreetMap). There are several potential bottlenecks on this stretch due to existing topological structures, namely an end of a plus lane (EoPL) – a (left) lane dedicated for fast vehicles – at around 330m-380m, an on-ramp (ON1) at around 1000m-1280m, an on-ramp (ON2) at around 2750m-3000m, and an off-ramp (OFF) at 3510m-3965m. The numbers are relative distances from the chosen route's origin which is the first detector (d1). Fig. 5.7b presents a simple schematic of

the road stretch. Regarding data, one year (2018) of 1-minute-aggregated speeds had been collected for the whole ring road. The data are provided by the National Data Warehouse (NDW), the Netherlands ndw. We have identified 778 traffic patterns that have congestion propagating to the selected corridor.

5.7.1 Detection of bottlenecks on a field-data pattern

One example pattern of traffic on the selected corridor is given in Fig. 5.8a. The objective is to detect the three activated bottlenecks. The top row shows the results with respect to the speed discontinuities detection. It can be seen that the response to the *inclined* kernel P_c (Fig. 5.8c) better highlights the locations and activation time of the three bottlenecks as compared to the response to the *vertical* kernel P (Fig. 5.8b). Notice that this advantage is more significant in cases that bottlenecks have high frequencies of disturbances (due to the direction on which we calculate the gradient). The second row shows the results obtaining from identifying bottleneck activation locations and time. Although different rectangles (i.e. bold lines) can be detected (see Fig. 5.8d), their representative lines lie on the corresponding bottleneck locations. By removing the lines or parts that lie outside of the congestion region as well as those that are too short, we obtain the final result as shown in Fig. 5.8f. This detection result is, qualitatively, the expected outcome given the speed pattern in Fig. 5.8a. Also, related congestion regions are sufficiently identified for each of the detected bottlenecks as shown in Fig. 5.8g. This example and many others in our experiment have further confirmed the efficiency of our proposed method in detecting relevant features of activated bottlenecks from traffic data.

5.7.2 Derived insights into the selected corridor

Fig. 5.9 illustrates the outcomes of applying the proposed method to the collected data. There are several interesting findings from the left figure, which is from raw data. First, the detectors d2, d9, and d8 are the most frequently detected activation locations. Their annual counts are more than 400 times which indicate on average they are activated every day. By associating with the topology information in Fig. 5.7, these locations are located near three topological disruptions. Particularly, d2 is just upstream of the end of the plus lane (EoPL) on the road stretch, d8 is at the end of the ON2's shoulder lane, and d9 is at the beginning of the weaving lane before the off-ramp (OFF). Notice that, the combination of ON2 and OFF potentially creates a weaving section which causes traffic congestion. The results also provide an overview of the variance of bottleneck activated locations. While almost all the activations are determined to trigger downstream of d2 in case of the EoPL bottleneck, there are more varieties with the ON2-OFF bottleneck. This might be expected as the EoPL is a kind of the lane drop bottleneck, and congestion usually occurs at the vicinity of the ending of the lane. Additionally, the next detectors - d1 and d3 - are quite far away from

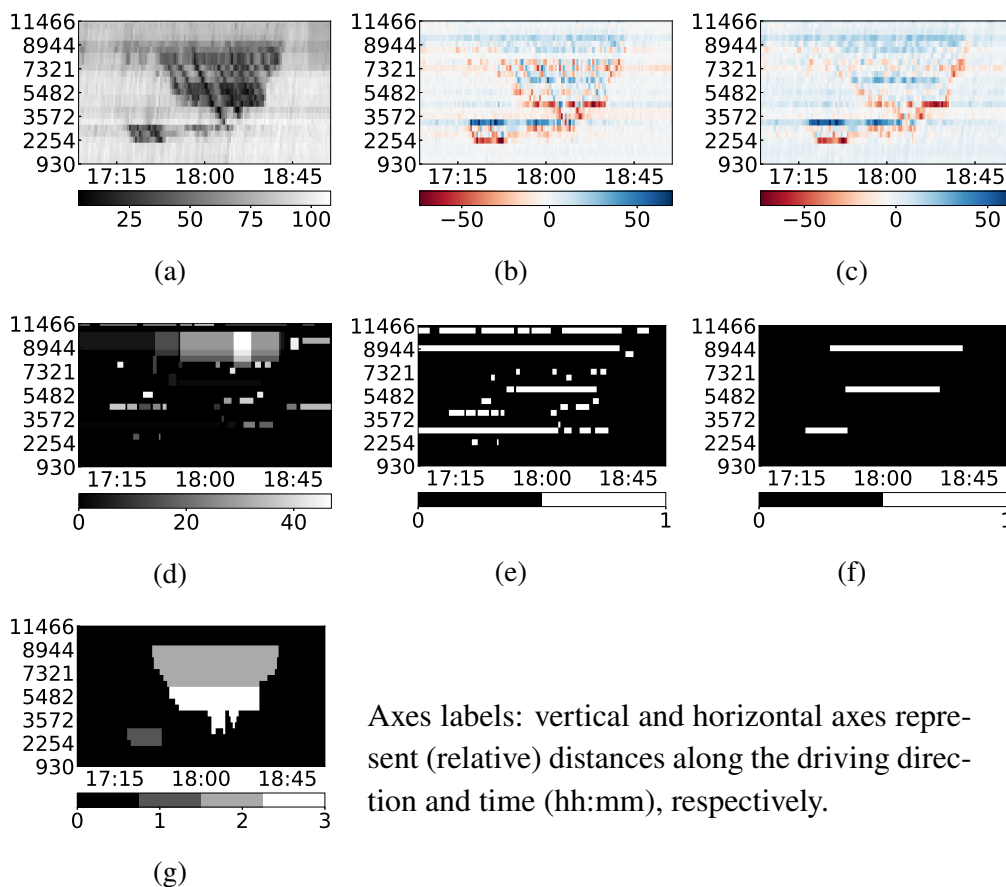


Figure 5.8: Intermediate results when applying our proposed framework to a traffic pattern. Top row - a pattern of traffic speed and its responses to different kernels (with respect to the detection of speed discontinuities in Sec. 5.5.1): (a) An example of traffic speed, (b) Response to the vertical kernel P , (c) Response to the *inclined* kernel P_c . Second row - identifying locations and activation time of (potential) bottlenecks (with respect to the proposed algorithm in Sec. 5.5.2): (d) rectangular regions, or *bold lines*, the corresponding locations and activation time before (e) and after (f) refinement. Bottom figure: (g) tracking congestion regions associating with different bottlenecks. Note that the underlying path of this pattern covers the entire selected corridor; therefore, its spatial extent is much longer than the distance of the corridor shown in Fig. 5.7. (For better visualisation of these plots, we refer the reader to the digital version.)

the EoPL which might explain the dominant detections of discontinuities at d2. In the case of ON2-OFF, the weaving traffic, namely trying to merge from the on-ramp and to leave the highway to the off-ramp, can create a lot of disturbances and trigger congestion when traffic is getting dense. Besides, congestion can also occur due to a high demand of ON2 merging traffic. Hence, traffic speed disruptions detected at d8 and d9 are understandable. Fig. 5.9 also shows much fewer amounts of potential bottleneck activations downstream of detectors d10, d6, and d1. This can be expected as there are no topologically potential bottlenecks downstream of these detectors. The detector d10 is the closest one to a physical disruption, however, it is located downstream of the off-ramp which seems to fall into the discharging areas. There are also noticeable amounts of activations at detector d4 which is just at the end of the first on-ramp's shoulder lane. The results suggest that this one does activate multiple times, although it is not as considerable as the downstream one.

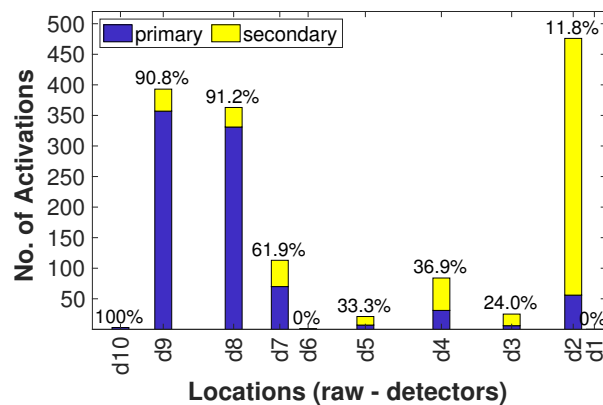


Figure 5.9: Activation locations of bottlenecks on the studied corridor during 2018. Bottleneck locations are associated with the most upstream detectors, with detector annotations are shown in Fig. 5.7b.

Regarding the *originality* of these bottleneck activations, we have two remarkable cases to discuss here. First, the on/off-ramp bottleneck were mostly the primary bottleneck. Approximately 90% of the detected activations originate at this location. The story is opposite in the case of the end-of-plus-lane bottleneck. Nearly 90% of the occurrences, it reacts to propagations of downstream congestion. The activation intervals of all detected bottlenecks are aggregated and depicted in Fig. 5.10c. The plot indicates strongly the two most outstanding bottlenecks on this corridor, namely the EoPL and ON2-OFF. The heat map is also in line with the significant correlation of these two, i.e. whenever the downstream is activated, most likely the upstream will be triggered. Their specific activation time is shown in Fig. 5.10a and Fig. 5.10b, respectively. These two bottlenecks active fairly often during morning and afternoon peak-hours, although the peaks are in the morning (in both bottlenecks). The primary activation counts (over time) are also depicted to reveal if there is any correlation with activation time. In this case study, the figures suggest no indication that the chance of getting primary activations differs with respect to morning or afternoon peak hours.

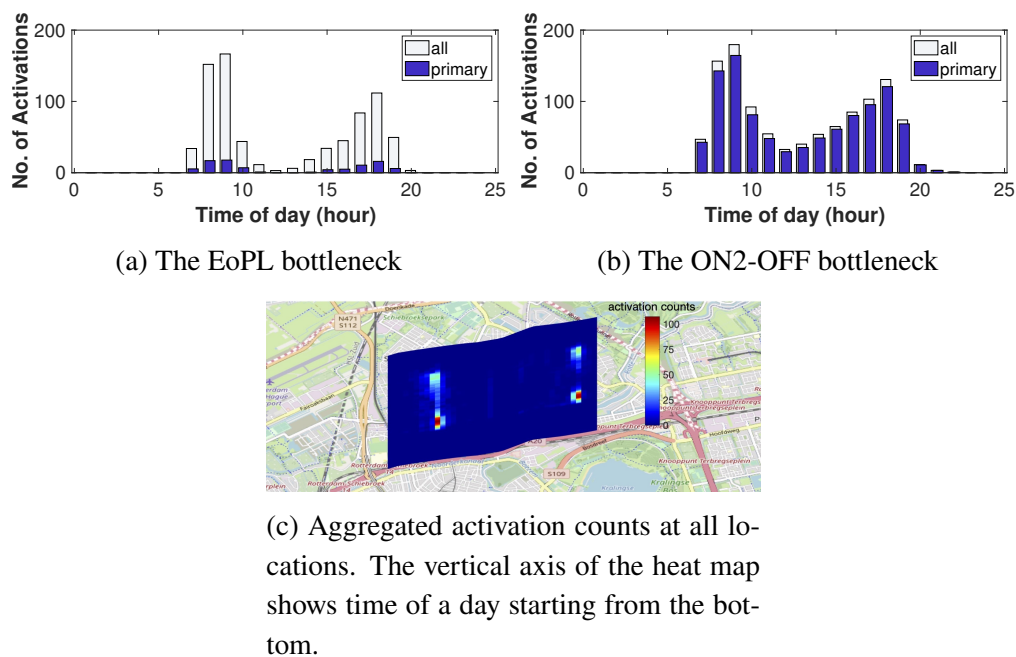


Figure 5.10: Activation time of bottlenecks on the studied corridor during 2018. Counts are aggregated over hours.

In conclusion, by automatically processing one year of traffic speeds, the proposed bottleneck detection method has found two most frequent bottleneck locations on the selected corridor, which is the EoPL and ON2-OFF. In addition, most of the time, the ON2-OFF bottleneck causes congestion and it, later on, triggers the EoPL. These findings suggest the majority of attention should be on the downstream location in order to mitigate the impacts of congestion and improve the quality of traffic on this corridor.

5.7.3 Time complexity

Fig. 5.11 shows the realised processing time of the proposed method. The complexity of a pattern is represented by the number of measurements from related loop-detectors. It appears that there is a linear correlation between processing time and pattern sizes, which is in line with the theoretical analysis in Section 5.6.3. In addition, the majority of patterns possess up to approximately 1.5×10^4 measurements and took less than 6 seconds to be processed. Hence, we can conclude that the method is efficient for offline processing bottleneck activations as well as potential for online applications.

5.8 Conclusion

This chapter has presented a method to automatically detect highway bottleneck activations in congested traffic patterns using image processing techniques. First, con-

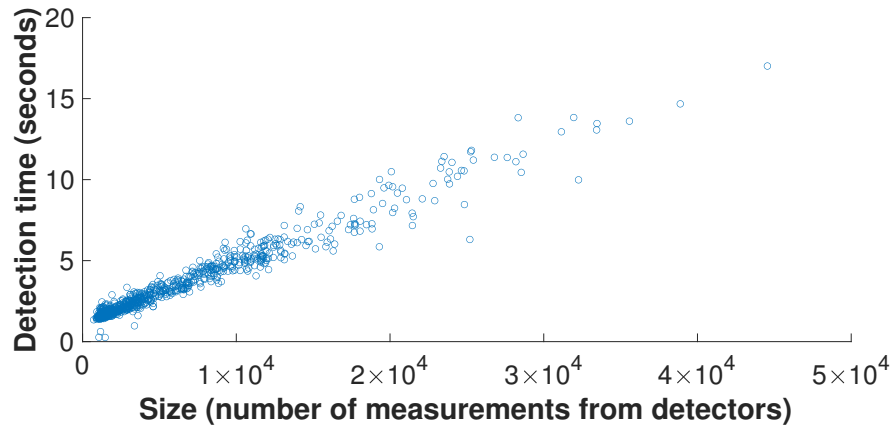


Figure 5.11: Processing time for detecting bottleneck activations.

gestion regions are identified using the Chan-Vese model, which is the active contour model without using edges. Second, the filtering kernel is constructed to detect speed discontinuities in raw traffic data, which subsequently gives approximate locations of bottlenecks. By calculating speed gradients along the direction of congestion waves, speed disruptions are efficiently highlighted. This applies to secondary bottleneck where their downstream traffic is congested; hence, assuring the detection thereof. In addition, information on the temporal dimension is incorporated to associate individual activating points at (the vicinity of) a location, which subsequently generate a comprehensive detection (represented as bold lines) of location and time of the related bottleneck. Third, congestion associated with the detected bottleneck is identified based on the results (overall congestion regions and bottleneck activation location and time) from the first two steps. Based on that, characteristics of associated bottlenecks can be calculated such as originality of congestion, i.e. primary or secondary source. The proposed method is investigated using both simulated data and real loop-detector data, based on which we have come to the following conclusions:

- Bottleneck activation locations can be determined efficiently by detecting speed discontinuities along congestion wave direction in loop-detector data.
- The accuracy of detected locations (by loop-detector data, and perhaps generally fixed-location data) depends on both bottleneck strengths and locations of loop detectors. The stronger a bottleneck is, the finer detector spacing is so as to determine where congestion saturates.
- Inherently from the above point, activated locations of weak bottlenecks, i.e. those with long accelerating distances, can be sufficiently determined under sparse spatial setting (500m to 1000m) of detectors.

For future studies, there are some opportunities for improving the method as following.

- The speed discontinuity kernel can be improved to not only account for congested waves but also incorporate free traffic and deceleration/acceleration area.

Given that every (spatio-temporal) traffic state is classified into congested or uncongested class, this is directly feasible from the proposed method.

- Network topologies can be incorporated in order to realize precise bottleneck locations. That means the proposed method in this paper acts as a rough detection of bottlenecks (by specifying related detectors).

Since the proposed method is automatic, it can process traffic patterns and extract bottleneck-related characteristics systematically. Hence, one can easily apply the method to large-amount of highway traffic data, which is increasing quickly over time, for conveniently mining relevant information. In addition, the association of congested traffic regions to the corresponding bottlenecks provides a precise way to measure or evaluate consequences of individual bottlenecks or combinations thereof on traffic flow. In practice, the automation and advanced consequence detection bring comprehensive tools for stakeholders such as traffic manager or policymakers to get valuable insights for important tasks such as bottlenecks evaluation and strategy assessment. As a result, the impacts of highway bottlenecks can be reduced or prevented to improve mobility on highways.

Chapter 6

Pattern Retrieval

In this chapter, we present pattern retrieval that essentially searches the database for patterns of interest. Arguably, the search-by-example is the most challenging one as the system needs to characterize given patterns and finds the most similar patterns in the database afterwards. For that purpose, we have identified two crucial elements involved, namely pattern representation and similarity measuring schemes. The former is concerned with how to represent patterns of congestion based on features that are learned previously. The latter applies relevant distance schemes on the representation obtained from the former step to measure the similarities between patterns. This results in their ranking of closeness towards a given pattern; hence, the most similar patterns are identified.

This chapter is based on the following paper that is (at the time of writing this thesis) in preparation for submission:

Nguyen, T. T., S. C. Calvert, G. Li, H. Van Lint. Pattern retrieval of traffic congestion using graph-based associations of traffic domain specific features.

6.1 Introduction

6.1.1 The necessity of congestion pattern retrieval

Innovation technologies have opened a new area of big data in many domains. On the one hand, these data provide many opportunities to gain valuable insights which are critically important for strengthening knowledge in any domain. On the other hand, vast amounts of data pose huge challenges in management and utilisation. In the field of transportation AVI - Automatic Vehicle Identification and FCD - Floating Car Data, many highly relevant traffic quantities, such as vehicular speeds or volumes, can now be collected in different degrees of spatial and temporal granularity, in part thanks to a variety of sensing systems such as induction loops. These data are beneficial for various purposes in road administration, industry and academia, including policy evaluation (Van Lint et al., 2005; Wang et al., 2006; Calvert et al., 2011), traffic management (Calvert et al., 2018), traffic modelling and simulation (van de Weg et al., 2018; Soriguera & Robusté, 2011; Vlahogianni et al., 2005).

The collected traffic data comprise of critical information for understanding many aspects of traffic, none so important as traffic congestion as a major nuisance and of major economic influence. Traffic congestion can be triggered at many different times and places due to various different reasons such as increasing travel demands at peak hours or incidents on roads, but also minor behavioural disturbances. Hence, we can expect numerous instances of congestion to exist in historical traffic data, which constitutes a valuable source of insights into traffic congestion. Arguably, it is advantageous to have a retrieval system that can identify similar congestion instances in historical traffic data. Such a retrieval system can pave the way for the development of many applications, such as traffic analysis and traffic prediction to analyse the recurrence of congestion. By checking if similar types of congestion occurred in the past, we can evaluate if a given congestion instance is recurrent or non-recurrent. In addition, if a type of congestion is recurrent, (possible) variations can be revealed based on analysing similarly occurring instances. A further step in this direction is to associate these similar instances with other relevant sources of information, e.g. incidents or topology, so that a more thorough understanding of certain types of congestion is derived. On the traffic prediction branch, the ability to extract similar patterns in historical data is a convenient tool for identifying (ir)regularities in traffic, which is highly valuable for predicting traffic states (e.g., (Lopez et al., 2017)). To the best knowledge of the authors, such a retrieval system for traffic congestion is currently lacking in transportation literature. This gap hampers the potential benefits of the collected traffic data, in particular, to obtain similar instances of congestion for different purposes and aid traffic analysis and prediction in congested traffic.

The dynamics of traffic involve both the spatial and temporal dimensions. Hence, congestion is effectively observed or evaluated by constructing two-dimensional maps of relevant data like speeds or flows. Such representation is visually intuitive and

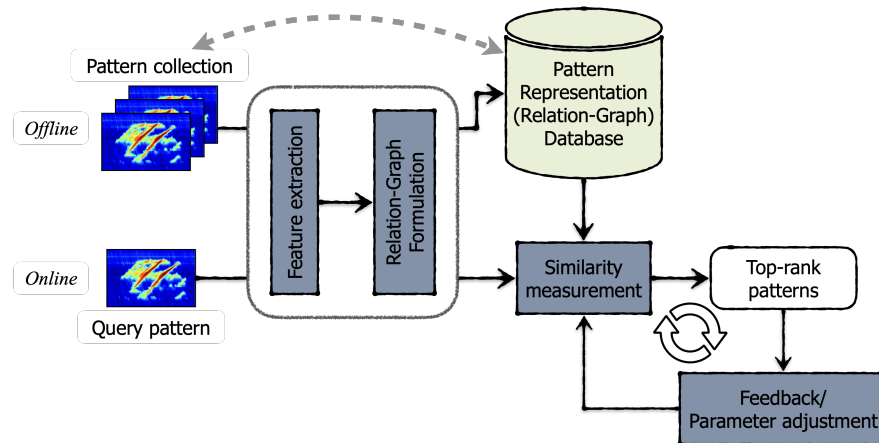


Figure 6.1: The overall framework for congestion pattern retrieval.

also reveals relevant insights into traffic phenomenon like wide-moving jams. This 2D representation of congestion motivates the creation of 2D maps, equivalently images, for traffic congestion occurrences, particularly at the corridor level. These maps are so-called congestion patterns. Subsequently, various techniques from computer vision will be applicable for different studies. For example, Nguyen et al. (2019) applied image segmentation methods to detect the two most common traffic phenomena, namely traffic disturbances and homogeneous regions (Helbing et al., 2009). The paper also shows that, compared to low-level image features (Bay et al., 2008), these traffic-specific elements can lead to more crisp clusters of congestion patterns. Since the constructed feature vector is a histogram of particular traffic regions, spatio-temporal relationships are not taken into account, which can hinder the performance when directly applying to pattern retrieval, in which (probably a limited number of) the most similar patterns are searched for from the dataset. Nevertheless, the benefits of representing congestion patterns as images motivate an approach focused for an image-like pattern retrieval system for congested traffic.

In general, there are two types of frameworks for image retrieval systems exist, namely text-based and content-based (Datta et al., 2008). In the former approach, each image is annotated with labels or so-called keywords when preparing a database. By specifying a particular keyword, the related patterns are easily identified and retrieved. The advance of this approach is that the retrieval mechanism is simple to implement. However, annotating images is usually done by hand, which is time-consuming and prone to errors due to numerous available items. Retrievals in the latter approach (see (Zhou et al., 2017) for a recent literature survey) are proceeded by providing an example image in advance. The retrieval system automatically extracts features from this image and searches for matches from available images in a database. Commonly used features are low-level information like colour, shape, texture. This approach can lead to an uninterpretable connection between the low-level visual features and their conceptual meanings. Therefore, this drawback limits the interpretations of retrieval outcomes.

To this end, this chapter proposes an information retrieval framework for occurrences

of traffic congestion and simultaneously aims to achieve a high level of interpretability of retrieval outcomes. In particular, each entire occurrence is captured and represented by a two-dimensional speed map, which acts as a fundamental object in the framework. Distinctive representation of a pattern is constructed at an abstract level using a graph-based association of traffic-specific regions. Traffic-domain features are essential for obtaining an abstract description of congestion patterns. The application of graphs preserves (possible) relations between components in a pattern, which essentially illustrate the overall structure. Moreover, to measure the similarity between two patterns, a matching method is formulated and parameterised regarding several observable characteristics of patterns. Accordingly, various expectations of patterns obtained from a query can be intuitively translated into proper configurations of those parameters.

6.1.2 General retrieval framework

Fig 6.1 illustrates the overall framework of our proposed retrieval system. There are two major components which are the so-called pattern representation and similarity measurement. The former consists of two sub-components: the feature extraction and the relation-graph formulation. The feature extraction determines and extracts important traits from a pattern, from which patterns can be compared. Afterwards, these individual characteristics need to be integrated into a cohesive instance that represents patterns. In this chapter, we propose employing graphs for that purpose due to their capability to preserve relations between regions in a pattern. The combination of these two sub-components leads to an abstract representation for any given congestion pattern. The similarity measurement component determines the degree of resemblance between two patterns. Its inputs include two relation graphs representing two patterns with its output being a similarity score.

In the offline stage, patterns are processed such that related features and their relation graphs are built and registered on a database. Patterns are retrieved in the online stage. If a retrieval outcome does not meet certain expectations, users can repeat the retrieval process to improve the result. There are two approaches for designing this iterative improvement, namely relevant feedback or parameter adjustment. In the former, users indicate whether individual obtained patterns are relevant or irrelevant. A proper method is implemented that takes users feedback and revises the retrieval result accordingly. In the latter approach, there the retrieval system is not automated. Based on their understanding of the retrieval framework, users can tune parameters to obtain desired patterns. Note that this last part is out of the scope of this chapter.

6.1.3 Chapter outline

The rest of the chapter is organized as follows. Section 6.2 describes the process of extracting relevant features and constructing the so-called relation-graphs as the repre-

resentation for congestion patterns. The measurement of similarity between two patterns is presented in Section 6.3. In Section 6.4, we describe an experiment for evaluating the proposed method. Finally, Section 6.5 concludes this study.

6.2 Pattern representation

To compare two patterns, we need to identify their representative characteristics, which we will refer to as *features*. Furthermore, it is important to develop a suitable mechanism for combining these features into a coherent structure. This allows the similarity between patterns to be computed and assessed effectively. This section respectively discusses these components in detail.

The feature extraction component aims to extract features that best describe an item, i.e. a congestion pattern in our application. They need to be selected such that patterns are well differentiated so that similarities are well measured. On the one hand, since congestion patterns are represented as images, low-level attributes, such as colour, shape, and texture, can be extracted to describe related patterns. On the other hand, as these images represent traffic congestion, they possess traffic phenomena that are well-acknowledged in the field. Traffic-related characteristics can provide a high-level semantic approach in formulating representative features for congestion patterns. In this chapter, we adopt the latter approach due to their ability to provide transparency and interpretation. Before going into the details, for the coherence throughout the chapter, we first define some common terminologies.

6.2.1 Terminology

Congestion patterns A congestion pattern represents congested traffic on a road stretch (or corridor) over a certain temporal period. In essence, it is a two-dimensional matrix of traffic states (such as speed, flow or density) where each value pertains to a traffic state on a road section and time period. To obtain such discrete values over location and time, we use the Adaptive Smoothing Method (Treiber & Helbing, 2002; Schreiter et al., 2010a) to map the (irregularly available) sensor data on equidistant grids. The result is equivalent to an (intensity) image, which is convenient for observing the traffic therein. In this chapter, the term *congestion pattern* refers to both the image representation and the underlying 2D matrix of traffic states.

Region (Area) A region (or an area) refers to a group of connected pixels in the equivalent image representation of a congestion pattern. The connection is either by the spatial or temporal dimension.

Traffic primitive (component) A traffic primitive (or component) refers to a region representing a specified traffic phenomenon, which will be introduced in the following sections.

6.2.2 Feature extraction

Congestion patterns can show instances of these widely-acknowledged phenomena, including wide moving jam, homogeneously heavy congestion, and traffic bottlenecks. These are visually observable in image patterns of congestion. The former two components are used and evaluated by Nguyen et al. (2019). By combining these two with the extents of congestion to formulate a feature vector, the authors derive different meaningful clusters of congestion patterns. The latter component is also a regular phenomenon of congestion as they are a common cause of traffic congestion. In this work, we incorporate and evaluate these three components as fundamental domain features of congestion.

Abstract primitives

Traffic disturbances Disturbances occur regularly in traffic and can be visualised effectively by spatiotemporal maps or traffic (image) patterns. They can emerge from a bottleneck where approaching vehicles try to synchronise with slow traffic therein. These disturbances can propagate further upstream and form wide-moving jams. Krishnakumari et al. (2017) proposed and successfully applied Active Shape Model to identify WMJs in image representation of traffic congestion. Since minor disturbances and wide-moving jams have similar shapes except for their spatial extents, this chapter further employs this method to determine those in congestion patterns.

The Active Shape Model technique (Cootes et al., 1995) describes a shape using a mean shape and its variations, which are obtained from a set of similar training shapes. Thus, given a new shape, the error of fitting the shape model to this shape can be used to identify or classify the shape. To obtain these shapes, pattern images are segmented using the Watershed transformation (Nguyen et al., 2019) into different traffic state regions. The boundaries of these regions are identified and clustered by the Active Shape Model for detecting traffic disturbances. We refer to the original paper (Nguyen et al., 2019) for further details.

Homogeneous congestion The homogeneous congestion represents the spreading of congested traffic over space and time with consistently low vehicular speeds. They are normally associated with strong bottlenecks or accidents where the mismatch between traffic demand and local supply is significant. The regions associating with homogeneous congestion are referred to as Demand-Supply element in (Nguyen et al., 2019), in which the authors propose a simple condition to detect their existence. In

this chapter, we adopt texture analysis for the identification of regions of homogeneous congestion.

Haralick et al. (1973) proposed deriving various texture features of an image using gray-level co-occurrence matrix (GLCM). It enables the calculations of different statistics to quantify/represent texture characteristics of the related image. This method has been one of the most popular approaches in image texture representation. Some widely used features are energy, contrast, homogeneity, entropy. Each number in the GLCM shows how frequent the related pair of intensities present in the related image with respect to a pre-defined (two-dimensional) offset.

A preliminary analysis suggests that the energy feature is most promising for identifying homogeneous regions in congestion patterns. In fact, energy is a measure of the homogeneity of an image. It is defined by Equation 6.1. The number of grey levels in a homogeneous region is expected to be low, which shifts the whole distribution to a small group of $p_d(i, j)$ ($p_d(i, j)$ represents the frequency of having the co-occurrence of intensities i and j at a certain distance d). The more homogeneous a region is, the higher the energy feature gets.

$$Energy = \sum_{i=1}^N \sum_{j=1}^N p_d(i, j)^2 \quad (6.1)$$

Traffic bottleneck Traffic bottleneck detection has a large body of literature that includes many approaches and methods. In this study, we adopt the framework in (Nguyen et al., 2021), which identifies bottleneck location and activation time from speed maps representing traffic congestion. Furthermore, the method also extracts the boundaries of upstream congestion regions, which are beneficial for further analyses. In principle, congestion regions are identified by applying the active contour model without edges (Chan & Vese, 2001) - a well-known image segmentation technique in computer vision. The model formulates congestion as foreground and free-flow regions as background in the corresponding segmentation problem. Bottleneck locations are detected by observing speed gradients along the direction of characteristic waves. Discontinuities (drops of speed at upstream) of traffic speeds are associated with possible bottleneck activations. Both primary and secondary bottlenecks can be identified successfully by this method. We refer to the original paper for a complete description of the framework.

Primitive characteristics

The objective of pattern retrieval is to localise patterns that best resemble a given pattern. The judgement of possible resemblances between two patterns can come from different perspectives of traffic observers.

We define two overall perspectives regarding the matching procedure of two congestion patterns, namely pattern structure and element detail. The former represents how the main traffic phenomena relate to each other within a congestion pattern. For example, a bottleneck causes congested traffic upstream, in which disturbances may have emerged and propagated upstream. The relevant traffic phenomena thus include the bottleneck and those disturbances—together they form the high-level structure. The latter perspective examines patterns *within the structure* by comparing details of different components, for example, the bottleneck severity and the frequency with which disturbances have emerged.

Regarding pattern structures, we differentiate between structural integrity and structural completeness. The former prioritises similarities in overall structure (size, area) as much as possible, and is tolerant of missing details in target patterns; whereas the latter focuses on target patterns having as much of all the constituent elements as possible regardless of their placements (locations, times) and the resulting overall structure.

With respect to element details, there is possibly an infinite number of characteristics that can be considered. To demonstrate the proposed framework, we therefore make a selection. Specifically, we consider the size of the patterns and the constituting elements, for which we look (i) at the relative proportions of elements in the related pattern, (ii) the absolute size of elements. Large relative proportions imply having similar patterns regardless of the absolute size, while absolute size focuses on actual sizes of patterns. As an example application, the former is preferable when looking for patterns with similar traffic phenomena, whereas the latter is more suitable when matching the consequences of congestion is important.

6.2.3 Relation-graph formulation

Feature representation combines attributes that are extracted from patterns in such a way that makes those patterns distinctive. In this chapter, we employ so-called *relation-graphs* as an alternative to a vector (a list) of features. The difference between this relation-graph and a feature vector representation is that a graph representation can describe not just the list of relevant features (encoded in the nodes) but also their relationships (encoded in the links between those nodes).

Specifically, a node (or vertex) in our relation-graph represents a traffic phenomenon from a limited set of so-called *traffic primitives*. We consider three such primitives and therefore three types of nodes in this study: these are the bottleneck (B), regions of homogeneous congestion (H) and disturbances (D), respectively. A node furthermore contains attributes describing the corresponding traffic primitive. In this study the main attribute of a node is *size*, in either absolute form [$\text{km} \times \text{hour}$] or relative form [proportion %]. A directed link (or edge) represents a spatiotemporal relation between primitives (i.e. traffic components). This relation indicates a possible causality based on the observation that the starting point t, x of one primitive is associated with the

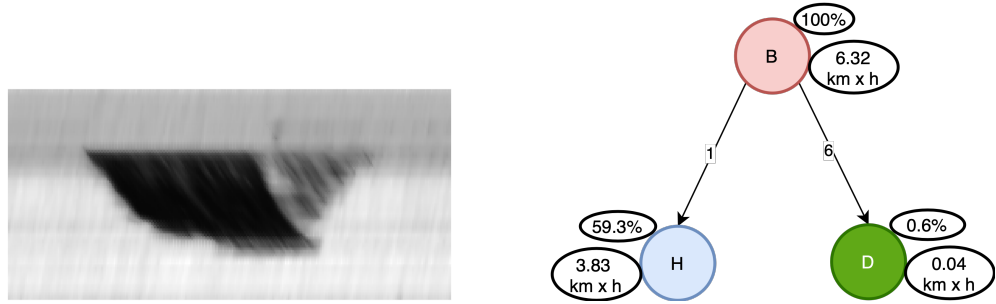


Figure 6.2: An example of relation graph: (a) a pattern of congested traffic at a bottleneck which causes heavily homogeneously congestion and later some small-scale disturbances, (b) its relation graph proposed by our method.

other primitive. For example, to represent many disturbances emerging from a single bottleneck, the corresponding relation-graph has an edge from the related bottleneck node to the related disturbance node, with the edge weight indicating the number of disturbances. This representation results in compact relation-graphs (see Fig. 6.2).

A formal definition of this relation-graph is described in Definition 6.1.

Definition 6.1 *The relation-graph representing a congestion pattern is an attributed, directed graph $G = (E, V, A)$. Descriptions of these sets are as follows.*

$$\mathbf{V} = \{v \mid v \text{ is a primitive}\}$$

$$\mathbf{E} = \{(v_i, v_j) \mid v_i \text{ (possibly) triggers } v_j\}$$

$$\mathbf{A} = (\tau, s^a, s^p, w) \text{ attribute set}$$

$$\tau : \mathbf{V} \rightarrow \{B - \text{bottleneck}, D - \text{disturbance}, H - \text{homogeneous congestion}\} \text{ node label}$$

$$s^a : \mathbf{V} \rightarrow \mathbf{R} \text{ absolute size of a node}$$

$$s^p : \mathbf{V} \rightarrow \mathbf{R} \text{ relative size, i.e. the proportion (\%), of a node}$$

$$w : \mathbf{E} \rightarrow \mathbf{R} \text{ number of instances of the connection representing by the corresponding edge}$$

Fig. 6.2 illustrates the principle with the traffic pattern (left) and the resulting relation-graph (right). The pattern shows traffic congestion at a bottleneck which is likely related to an incident. At the onset, traffic is heavily homogeneously congested. After some time, a few (minor) disturbances emerge before traffic regains free-flow condition. The corresponding relation-graph is constructed by identifying the three main elements in this pattern. These include the bottleneck - B node, homogeneous congestion node - H node, and disturbances - D node. Edges are associated with w . Specifically, the link (B-H) has a weight of 1 to represent 1 homogeneous region as shown in the pattern, whilst the link (B-D) has a weight of 6 that shows the number of disturbances detected. Furthermore, each node consists of attributes, namely absolute size and proportion.

6.3 Similarity measurement

The previous section shows how to construct an abstract and intuitive relation graph for a pattern of congestion. This section describes how similarities between congestion patterns are measured based on those graphs. Overall, (in-exact) graph matching is adopted, and a similarity function, which measures the resemblance between any pair of nodes in two respective graphs, is proposed. This function reflects several aspects of a pattern, including the similarities in the structure, the proportions and frequencies of any extracted components. Details are given in the following paragraphs.

6.3.1 Brief overview of graph matching

By presenting congestion patterns using relation graphs, the measurement of similarities is transformed into graph similarity or so-called graph matching. There are two main categories in graph matching, namely exact graph matching and inexact graph matching (also known as error-tolerant graph matching) (Conte et al., 2004; Foggia et al., 2014; Riesen, 2015; Emmert-Streib et al., 2016). The former is strict in matching two graphs with respects to mapping their nodes or edges. This category is mainly intended for matching identical graphs. Meanwhile, the latter category is more flexible and allows differences in node/edge/subgraph mappings. In principle, these differences are tolerated with certain penalties when comparing graphs. This feature makes the second category more applicable to real-world applications where exact matching is not always guaranteed (if not rarely). In our application, matching relation-graphs representing congestion patterns falls into the second category.

A graph matching is formulated as an optimisation problem, in which the cost of a matching function is generally defined as shown in Equation 6.2 (rewritten from (Foggia et al., 2014)). Note that, in inexact mapping, some nodes or edges of one graph might not have matches from the other graph. To formally describe this, a special so-called null node ε is introduced. Accordingly, the mapping f is annotated as $f : V_A \mapsto V_B \cup \{\varepsilon\}$. It is injective for nodes in V_A that is not mapped to ε . For such nodes, the cost is called replacement cost C_R^N . Mapping a node to ε is reasonably seen as the deletion of that node, and the related cost is called deletion cost C_D^N . Besides, edges are also needed to be mapped. Two similar types of mapping, i.e. replacement and deletion, are relevant to edge mapping and are evaluated by the cost functions C_R^E, C_D^E , respectively. Note that these individual cost functions are specialised, which means their definitions depend greatly on specific applications.

$$\begin{aligned}
C(f) = & \sum_{\substack{v \in V_A \\ f(v) \neq \varepsilon}} C_R^N(v, f(v)) + \sum_{\substack{v \in V_A \\ f(v) = \varepsilon}} C_D^N(v) + \sum_{\substack{v' \in V_B \\ f^{-1}(v') = \varepsilon}} C_D^N(v') + \\
& \sum_{\substack{e = (v_1, v_2) \in E_A \\ e' = (f(v_1), f(v_2)) \in E_B}} C_R^E(e, e') + \sum_{\substack{e = (v_1, v_2) \in E_A \\ e' = (f(v_1), f(v_2)) \notin E_B}} C_D^E(e) + \sum_{\substack{e' = (v'_1, v'_2) \in E_B \\ (f^{-1}(v'_1), f^{-1}(v'_2)) \notin E_A}} C_D^E(e')
\end{aligned} \tag{6.2}$$

Various approaches have been proposed for graph matching by reformulating an optimisation problem on the cost function $C(f)$ such as graph edit distance (Bunke, 1997; Gao et al., 2010), graph kernels (Gärtner et al., 2003), iterative methods (Blondel et al., 2004; Zager & Verghese, 2008). We refer to (Foggia et al., 2014; Emmert-Streib et al., 2016) for an in-depth survey of these approaches. Our work is motivated by the iterative approach. In principle, the similarities between nodes consider not only the two nodes but also their neighbour nodes. Hence, this approach, to some extents, combines the notations of different individual cost functions (Equation 6.2) into one similarity function.

We propose a two-phase algorithm for measuring the similarity of two congestion patterns based on their relation-graphs. Firstly, the similarities of all possible pairs of nodes between the two graphs are calculated. Secondly, the total similarity score of mapping all available nodes is optimised. The obtained score represents how similar the two patterns are. The following subsections describes these two terms in detail.

6.3.2 Phase 1: Nodes similarity

The similarity between two nodes (source nodes) is measured in a recursive way as motivated by Zager & Verghese (2008). Specifically, the similarity of subsequent nodes recursively contributes to the similarities score of their source nodes. Unlike in (Zager & Verghese, 2008), where scores from all possible pairs of nodes are accumulated, our proposed method only considers those from the best mapping between subsequent nodes.

Similarity score for nodes

The overall similarity score between two nodes, $n_A \in V_A, n_B \in V_B$ from G_A, G_B respectively, is formulated as shown in Equation 6.3. The first part of the R.H.S, S_0 , measures the similarity that is based intrinsically on their attributes (regardless of their neighbour nodes). The second part represents the accumulation of similarities from their subsequent nodes. Here, the parameter θ_i regulates how much of subsequent nodes similarity attributes to the similarity of two source nodes. Note that the contribution of subsequent node similarities is, to some extent, equivalent to the similarity of matching the corresponding links (which is related to function C_R^E in Equation 6.2).

$$S(n_A, n_B) = S_0(n_A, n_B) + \theta_i \times \min(S_0(n_A, n_B), \arg \max_{f: C_A \rightarrow C_B} \sum_{c_i^A \in C_A} S(c_i^A, f(c_i^A))) \quad (6.3)$$

where C_A, C_B represents the sets of subsequent nodes of n_A, n_B , respectively.

Similar to the overall cost defined in Equation 6.2, the base similarity S_0 captures several possibilities of node matching, which depend on whether both nodes are in the original graphs. Accordingly, two similar evaluations need to be defined, namely the so-called *replacement* - $S_R(n_A, n_B)$ and *deletion* - $C_D(n)$. Equation 6.4 summarises these cases.

$$S_0(n_A, n_B) = \begin{cases} S_R(n_A, n_B), & \text{if } n_A \neq \varepsilon, n_B \neq \varepsilon \\ -C_D(n_A), & \text{if } n_B = \varepsilon \\ -C_D(n_B), & \text{if } n_A = \varepsilon \end{cases} \quad (6.4)$$

There are two cases when matching two non-null nodes regarding whether they represent the same primitive type. If these nodes are different types, their mapping is equivalent to two deletion operations (see Equation 6.5).

$$S_R(n_A, n_B) = \begin{cases} M(n_A, n_B), & \text{if } \tau(n_A) = \tau(n_B) \\ -C_D(n_A) - C_D(n_B), & \text{if } \tau(n_A) \neq \tau(n_B) \end{cases} \quad (6.5)$$

The previous setup leads to defining two basic functions, i.e. $M(n_A, n_B)$ and $C_D(n)$. Choices for these functions are specialised with respects to specific applications. In our proposed framework, we formulate these functions with respects to the selected attributes associated with nodes/edges in relation-graphs. Also, these functions are parameterised by utilising certain parameters. The objective is to inject different perspectives when looking for similar characteristics from congestion patterns.

Balancing similarity and differences

Our proposed function for measuring similarity between two commonly labelled nodes accounts for both the resemblance between their attributes and the importance of their difference. For that, the designed function includes both their overlapping size and the size of the referenced node. The former acts as a proxy to the similarity of the two nodes. The latter is used to compensate for the difference (if any) between the two nodes. The first node is selected as a referenced node in our set up. The parameter θ_g regulates the scales of these two terms (see Equation 6.6).

The detailed similarity between two nodes is measured based on the overlapping size. Note that a different function is possible when different properties are used for node attributes. On the other hand, the unmatched size is also taken into account as this

assists in ranking the closeness of different pairs of nodes. In particular, a logistic function is formulated to translate the size difference (in terms of proportions to the total size) to a number (i.e. weight) that scales the overall similarity. The contribution of this difference is regulated by the parameter $\theta_s \geq 0$ (see Equation 6.6). In addition, the difference in the occurrences (w) of the two nodes is also dealt with. A 'virtual node' n_E , with relevant features, a (see Equation 6.13) and w , is created as shown in Equation 6.11. The underlying idea is to apply deletion cost $C_D(n_E)$ to the occurrence difference when matching two nodes. Parameter $\theta_w \geq 0$ regulates the tolerance of this difference.

$$M(n_A, n_B) = (1 - \theta_g) * [2 \times w_{min} \times a_{min} \times \mathcal{L}_{\beta_1, \beta_0}(\theta_s, \frac{\Delta a}{\sum a}) - C_D(n_E)] + \theta_g \times 2 \times w(n_A) \times a(n_A) \quad (6.6)$$

where,

$$\text{Common size} \quad a_{min} \quad = \min(a(n_A), a(n_B)) \quad (6.7)$$

$$\text{Size difference} \quad \Delta a \quad = |a(n_A) - a(n_B)| \quad (6.8)$$

$$\text{Total size} \quad \sum a \quad = a(n_A) + a(n_B) \quad (6.9)$$

$$\text{Logistic function} \quad \mathcal{L}_{\beta_1, \beta_0}(\theta, x) = 1 - \frac{1}{1 + e^{\beta_1(\theta x) + \beta_0}} \quad (6.10)$$

$$\text{Occurrence-difference node } n_E \quad \begin{cases} a = \begin{cases} a(n_A), & \text{if } w(n_A) > w(n_B) \\ a(n_B), & \text{if otherwise} \end{cases} \\ w = f_{\beta_1, \beta_0}(\theta_w, \Delta w) \end{cases} \quad (6.11)$$

$$\text{Occurrence difference} \quad \Delta w \quad = |w(n_A) - w(n_B)| \quad (6.12)$$

$$\text{Size selection:} \quad a(n) \quad = \begin{cases} s^a(n), & \text{for absolute size, i.e. area (km} \times \text{minute)} \\ s^p(n), & \text{for proportion (\%)} \end{cases} \quad (6.13)$$

Node deletion cost

As overlapping size are used for attributing commonly labelled nodes, the cost of deleting a node can be justified by its size. A parameter θ_d is introduced here to regulate how much penalty is applied for not finding a match for a node. Equation 6.14 give a definition of this cost.

$$C_D(n) = \theta_t \times a(n) \quad (6.14)$$

Table 6.1 summarises all the parameters and their meanings in customising a similarity measurement between any pair of nodes.

Table 6.1: Parameters for customising similarity measurement between relation-graphs

Parameter	Description
θ_s	Penalise size difference
θ_g	Regulates the trade-off between node size match (maximised when $\theta_g = 0$) and node type match (maximised when $\theta_g = 1$)
θ_d	Penalise node type difference, therefore, regulates the tolerance of having unmatched nodes
θ_w	Penalise the differences in frequency attribute: whether to focus on overall structure or details
θ_i	Regulate the contribution of subsequent-node similarities to the matching of two source nodes

6.3.3 Phase 2: Nodes mapping

Given two relation graphs that represent two congestion patterns, the previous section shows how to measure the similarity between any pairs of nodes therein. This section describes how to come up with a similarity score at the pattern level.

To evaluate how the two patterns match, we formulate the problem as an assignment problem which finds the so-called perfect matching between nodes from the two graphs. That assignment maximises the total scores from all pairs of matched nodes under the condition that one node is matched with exactly another one. This perfect matching (once found) is considered the best mapping between the two source nodes. The corresponding total score then indicates the similarity between the two patterns. An illustration of our assignment problem is depicted in Fig. 6.3. A complete bipartite graph is constructed to show all possible mapping of nodes from two graphs. The weight of each edge is associated with the similarity score of corresponding nodes. The solution of pattern mapping is the perfect matching with the maximum total edge weights. Equation 6.15 formulates this assignment approach in mathematical term.

$$S(p_A, p_B) = \arg \max_{f: \Omega_A \rightarrow \Omega_B} \sum_{n \in \Omega_A} S(n, f(n))$$

where,

$$\begin{aligned} \Omega_A &= V_A \cup \{\varepsilon, \dots, \varepsilon\} \\ \Omega_B &= V_B \cup \{\varepsilon, \dots, \varepsilon\} \\ |\Omega_A| &= |\Omega_B| = |V_A| + |V_B| \end{aligned} \tag{6.15}$$

The assignment problem is solved by applying the well-known Hungarian algorithm (also known as the Kuhn-Munkes algorithm), which was developed by Kuhn (1955). It has polynomial complexity, in particular, $\mathcal{O}(n^3)$.

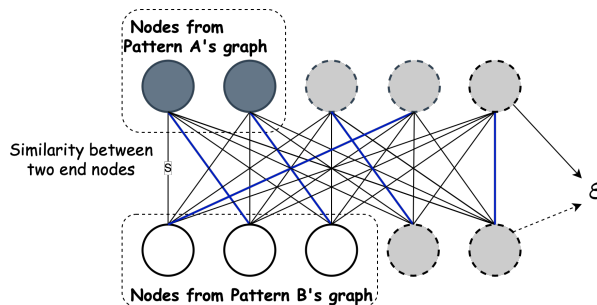


Figure 6.3: An illustration of how to formulate the pattern matching as assignment problem between their node sets. Edges' weights are the similarities between the corresponding end nodes using Equation 6.3. A feasible assignment is highlighted in blue colour, in which one node is exactly matched to another node.

6.4 Experiment results and discussion

In this section, we demonstrate the ability of the proposed retrieval framework in retrieving similar patterns from a collection of traffic congestion patterns. The analysis includes three aspects. First, some exemplary queries are conducted, and their performances are investigated with respect to the corresponding obtained patterns. Second, we discuss the impacts of tuning the parameters (in Table 6.1) for reflecting different perspectives on similarities between patterns. Third, we consider the computational complexity as well as its implication in applying to large datasets of the proposed method. Details are in the following sections.

6.4.1 Data & parameter settings

To evaluate the proposed method, we have selected a corridor on the ring of Rotterdam, which is one of the busiest roadways in in the Netherlands. Fig. 6.4 shows a broad view of the road. It is approximately four kilometres long, and comprises several active bottlenecks. These bottlenecks and downstream bottlenecks have caused much recurrent traffic congestion, therefore, it is a suitable choice for evaluating our proposed framework.

Speed data are provided by the National Data Warehouse (NDW), the Netherlands ndw, in which each measurement is a one-minute aggregation of speed surpassing the related induction-loop detector's implemented location. To have a better view of resulting traffic, we apply the ASM method (Adaptive Smoothing Method) (Treiber & Helbing, 2002; Schreiter et al., 2010a) to estimate speeds at finer resolutions both spatially and temporally, namely 100 meters by 30 seconds. We have processed data from the entire 2018 to obtain 778 patterns, which constitute the collection of traffic congestion patterns for evaluating our proposed method.

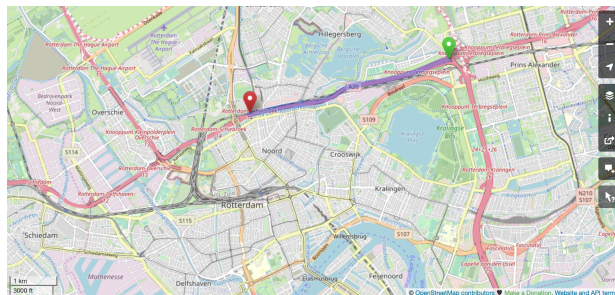


Figure 6.4: A broad view of the selected corridor in the experiment. The image is taken from Open Street Map (and reproduced from (Nguyen et al., 2021))

For the similarity measurement, the settings for all parameters are given in Table 6.2. By setting θ_t , θ_s , and θ_w to 1, the total differences in type, size, and frequency, respectively, are fed to the logistic function to measure related penalties. As θ_i is set to 1, similarities from subsequent-nodes are accumulated to the corresponding president nodes. This, to some extent, takes pattern structure into consideration. Therefore, we set θ_g to 0 for simplifying the base similarity function. This leads to full assessment of related attributes when matching two nodes. Chosen values of β set the changing point of the corresponding logistic function at the middle of input ranges. Note that there are no strict regulations in selecting these parameters. The presented settings are one of many possibilities and have lead to good results in our experiment.

Table 6.2: Parameter settings in the conducted experiment

Parameter	Value
θ_s	1
θ_g	0
θ_t	1
θ_w	1
θ_i	1
<i>logistics</i> \mathcal{L}	
(β_1, β_0)	(10, -5)

6.4.2 Retrieval results

To demonstrate the feasibility of the proposed relation-graph in the retrieval application, we analyse some exemplary queries, namely for single disturbance, stop-and-go congestion, homogeneous congestion and a mix of these. These are typical patterns of congestion that are commonly observed in traffic data (Helbing et al., 2009; Nguyen et al., 2016; Krishnakumari et al., 2017; Nguyen et al., 2019). Finding their occurrences provide meaningful information for various purposes. For instance, to analyse

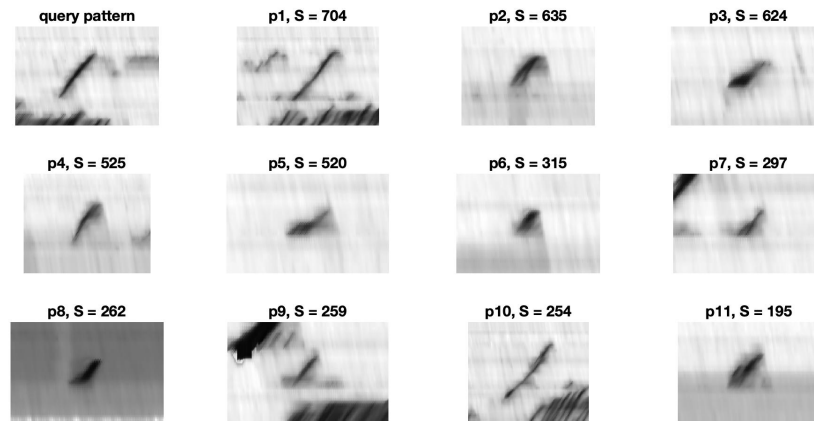


Figure 6.5: The 11 most similar patterns returned from searching for a moving disturbance (shown in the top-left pattern). Patterns are shown in the same resolution, hence, their size differences can be relatively shown. Note that, regions of congestion at the edges of some patterns should be ignored because they are the results of cropping out the patterns, i.e. they are not included as (main) content of the patterns. Besides, similarity scores are given as S for each of the patterns.

how repetitively these types of congestion occur, and possible variations therein. This leads to a more thorough understanding of popular types of congestion. Another application for this is traffic model development, specifically model evaluation. Different scenarios, which are derived from similar patterns, can be tested with respect to the occurrences of certain congestion patterns.

Single disturbance retrieval

Figure 6.5 shows an example of retrieving patterns representing a single disturbance. The implemented framework successfully returned patterns representing small disturbances as indicated in the query pattern. Note that, as disturbances are one of the nodes in the relation-graph, this retrieval outcome is a direct result of the extracting method. Regarding the order of these patterns, some might seem more similar than those in higher-ranks. For example, the pattern p4 seems more resembling the query pattern than the above two patterns (in the order list). The reason is that by choosing areas as an attribute, we have reduced two dimensions, i.e. spatial and temporal, down to only one. Therefore, introducing propagating lengths as attributes for disturbance nodes could fine-tune the results further. Nonetheless, in this work, we use the same attributes for all the nodes to simplify the graph and focus on demonstrating the feasibility of the proposed approach. We leave this enhancement for future work.

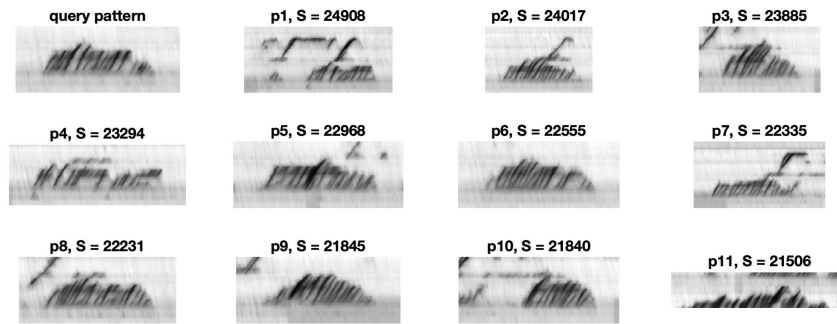


Figure 6.6: Retrieval results of stop-and-go congestion.

Stop-and-go congestion retrieval

Stop-and-go traffic waves is another common type of congestion where multiple disturbances occur over time. An example of retrieving such patterns is shown in Fig. 6.6. In the query pattern, a bottleneck is activated, from which many disturbances emerge. All the obtained patterns represent the same traffic phenomena. By detecting both the primary bottlenecks and probably the minor upstream secondary bottleneck, along with multiple disturbances, the resulted relation-graphs are effective for locating patterns with the same topology as in the query.

Homogeneous congestion retrieval

An example of retrieving homogeneous congestion is illustrated in Fig. 6.7. The given pattern represents significantly slow traffic upstream of a bottleneck (probably due to incidents like accidents). Hence, the two most important components of the corresponding relation-graph for this pattern is a bottleneck node and a homogeneity node. Overall, the obtained patterns do represent the main phenomenon. This effective retrieval is a direct outcome of the extracting method for domain-specific features.

Note that the shapes of homogeneous areas in the obtained patterns are not necessarily identical to that of the one in the query pattern. This is explained by the fact that the currently chosen attribute includes only sizes.

Complex congested traffic retrieval

Fig 6.8 illustrates an attempt to retrieve large scale congestion patterns. The query pattern consists of various types of traffic jams, including disturbances that occur fairly frequently, multiple bottleneck activations, and a homogeneous congested area. Many obtained patterns can cope with these complications in the input pattern, meaning they have different activations of bottlenecks that cause dense stop-and-go traffic. Some of them show homogeneous regions. Regarding the overall structure, several patterns (for instance, p1, p2 or p3) represent two clusters of disturbances that are potentially due to the activations of two primary bottlenecks.

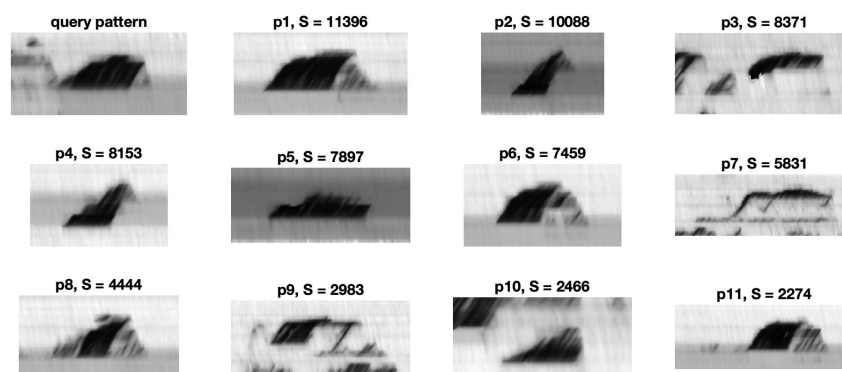
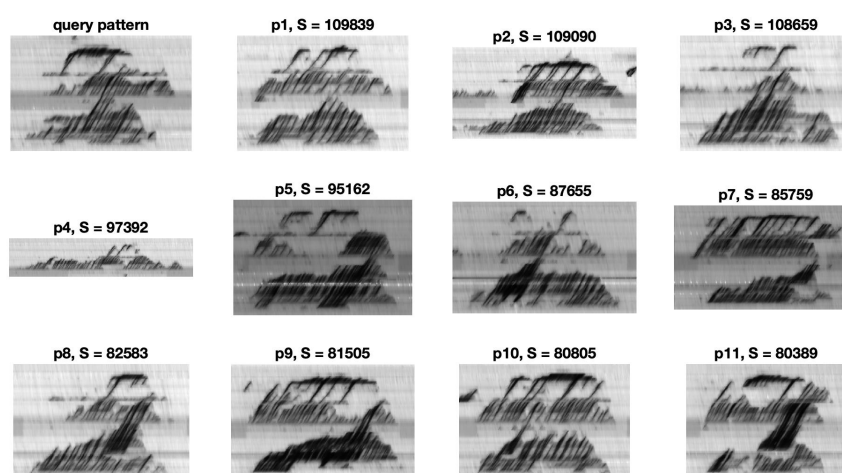


Figure 6.7: Retrieval results of homogeneous congestion.

Figure 6.8: Retrieval results of *meta* congestion.

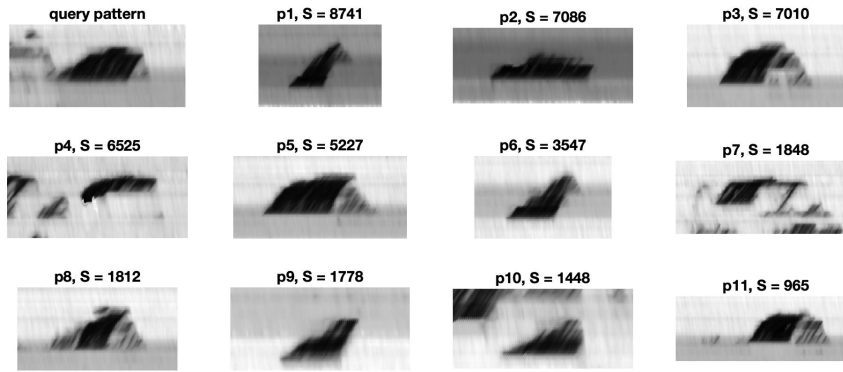


Figure 6.9: Another retrieval result of the homogeneous congestion in Fig. 6.5) with different parameter $\theta_s = 2$.

6.4.3 Parameter impacts

In this section, we analyse how modifying parameters can change similarity scores, hence, alter the ranks of obtained patterns. This is relevant for revising retrieval results in case a result is not as expected. The list of all parameters is shown in Table. 6.1.

Size penalty θ_s

The parameter θ_s penalises the difference in sizes between two matched nodes. Therefore, it regulates how important is searching for nodes of similar types. To demonstrate this, we modify $\theta_s = 2$ for the retrieval in Fig. 6.7. This modification enforces a stricter condition on the sizes of matching nodes. The corresponding result is shown in Fig. 6.9. Overall, the similarity scores of returned patterns decrease. The order of patterns consists of various changes such as the promoting of p1 (from the 2nd to the 1st place) and p5 (from the 1st to the 5th). In addition, new patterns are also moved forward, such as p10.

Weight penalty

The parameter θ_w controls the frequency of a component's appearances. This is mostly relevant to disturbances in stop-and-go traffic patterns. By increasing or decreasing this parameter, the outcomes are adjusted to be against or in favour of the differences in the frequencies of disturbances.

Fig. 6.10 demonstrates the impact of increasing θ_w on the searching made in Fig. 6.6. Even though there is not much (overall) difference compared to the previous result, this new result shows several changes in the order. The new ranking promotes those patterns with more similar numbers of disturbances as in the example pattern. The overall similarity scores are smaller due to the stricter condition of occurrence frequencies.

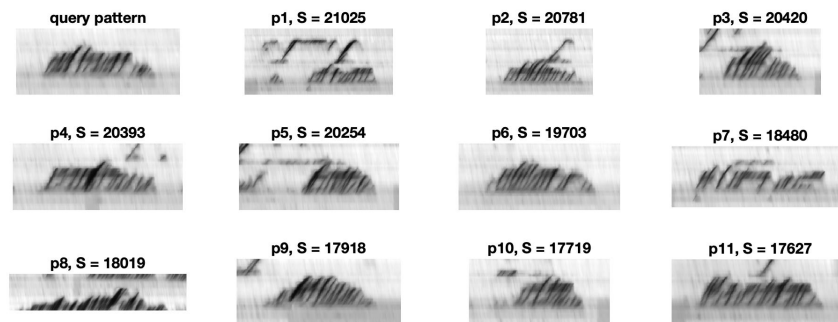


Figure 6.10: Another retrieval result of the stop-and-go congestion in Fig. 6.6 by increasing θ_w to 3.

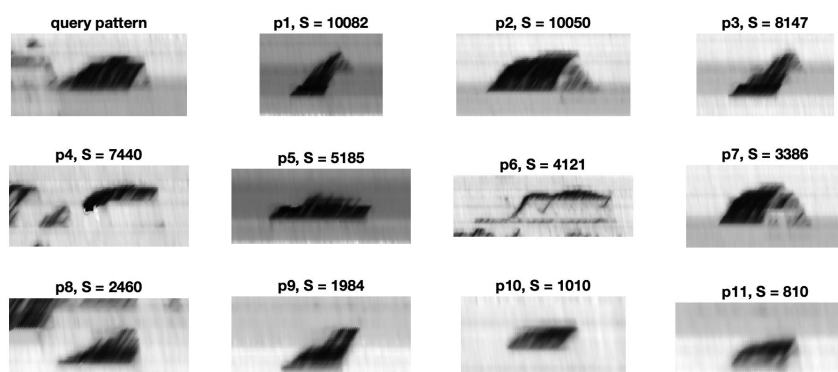


Figure 6.11: Another retrieval result of the homogeneous congestion in Fig. 6.7 by increasing θ_d to 2.

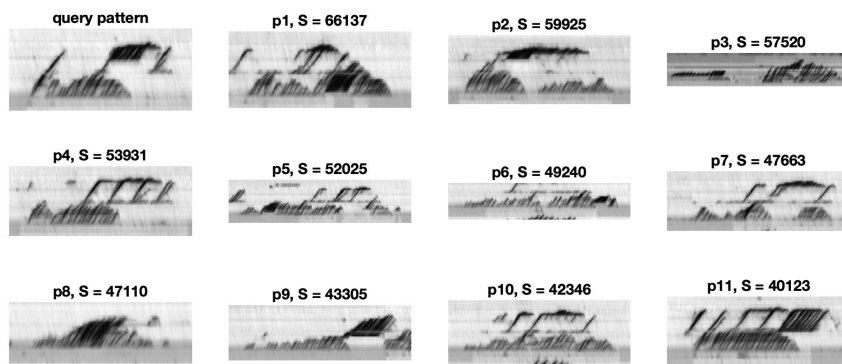
Unmatch penalty

There may be unmatched nodes from two relation graphs. How much this decrease the similarity score is regulated by the parameter θ_d . By lowering this parameter, users opt for finding the completion of the components in the query pattern, and at the same time tolerate the existence of extra components in the target patterns. Similarly, increasing θ_d aims for the compact of target patterns with respects to the given pattern.

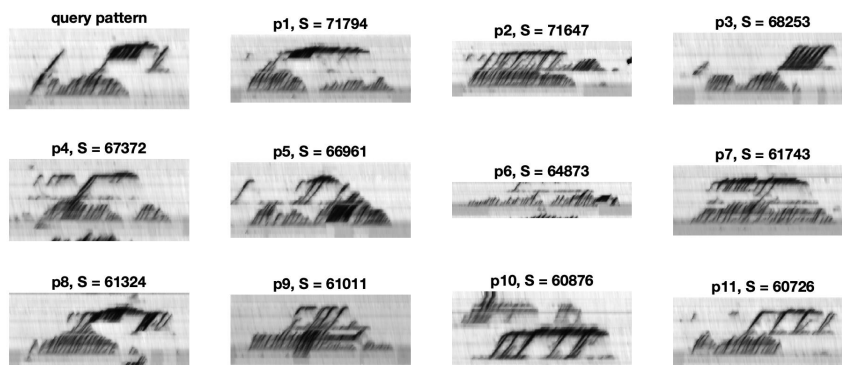
An example of the effect of increasing θ_d is shown in Fig. 6.11, which is a modified retrieval of the one in Fig. 6.7. Since θ_d has a higher value, those patterns with a more compact representation of homogeneous regions, less other extra regions, are advanced in the ranking list.

Structural integrity

The parameters θ_g, θ_i are designed to promote the matching of pattern structures. A demonstration of their use is illustrated in Fig. 6.12. Fig. 6.12a shows the example in which both similarities of pairs of matched nodes and their subsequent nodes are relatively important. On the other hand, by setting $\theta_g = 0.7$, the importance of having



$$(a) (\theta_s, \theta_g, \theta_d, \theta_w, \theta_i) = (1, 0, 1, 1, 1)$$



$$(b) (\theta_s, \theta_g, \theta_d, \theta_w, \theta_i) = (1, 0.7, 1, 1, 1)$$

Figure 6.12: The effect of the parameter θ_i .

the same structure becomes higher while that of node similarities are reduced. The obtained patterns in Fig. 6.12b demonstrate the effect of this change. Differences between components of the obtained patterns and those in the example patterns are more tolerant. As a result, some good similar patterns are advanced to the top list, e.g. p1, p3, p4, p8. Note that, increasing θ_i leads to low importance levels of node features. This, therefore, can result in patterns that are quite different from the query example despite sharing a common structure.

6.4.4 Time complexity

The processing time in the proposed method is spent mainly on relation-graph construction and graph-similarity measurement. Regarding the former, relation-graphs of all congestion patterns in the database are pre-processed and registered in advance. Hence, at the moment of retrieval, only the example pattern needs to be parsed. The processing time depends (almost) linearly on the size of the corresponding congestion region (or pattern) as shown in Fig. 6.13. In addition, the majority of patterns have sizes of approximately under 1000 ($\text{km} \times \text{minutes}$) and take around 60 seconds to build their relation-graphs.

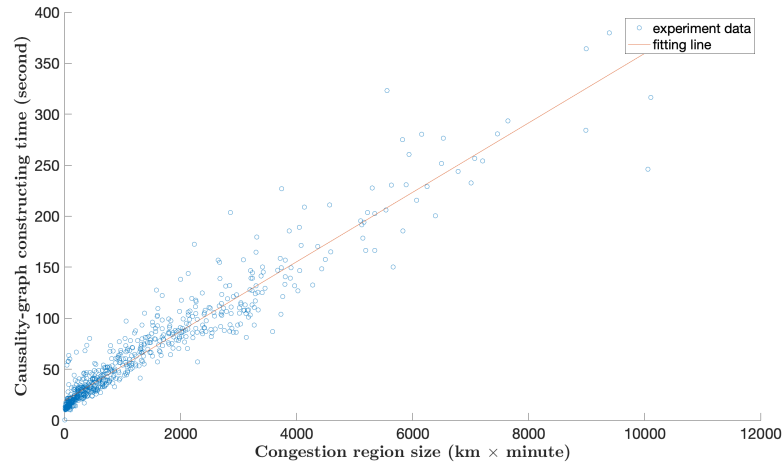


Figure 6.13: The constructing time of causality-graphs of all the patterns in the experiment data.

Fig. 6.14 illustrates the computation of relation-graph similarity. This includes times for matching every single pairs as shown in Fig. 6.14a and the total retrieving time in the experiment dataset shown in Fig. 6.14b. It can be expected that the time complexity of the proposed matching method for relation graphs is polynomial w.r.t graph size (measured in the total number of nodes and edges). From a close examination of Fig. 6.14b, it takes less than one minute to retrieve similar patterns for a pattern of up to 30 nodes plus edges in its relation-graph. However, the waiting time can be long for large-scale patterns or a collection of numerous patterns. Therefore, to scale up the proposed method to larger datasets, further improvements are necessary. One approach is to narrow down the searching space by some quick pre-processing. For example, as suggested by Fig. 6.14a, when retrieving for small-scale patterns, a (computationally) fast filter can be applied to keep only patterns with small numbers of nodes in their relation-graphs. Another approach is to employ more computational power where measuring similarities can be done in parallel processing, hence, reducing the responding time.

6.4.5 An opportunity for semantic retrieval

The relation-graph essentially represents traffic patterns in an abstract form that comprises nodes representing traffic phenomena. Therefore, this graph can be used as an alternative representation approach for describing expected patterns for retrieval. This way of searching has the advantage that users have high control of what to expect from the returned patterns, therefore, it is not limited by a given pattern. Take the following expectation as an example: *“bottleneck that causes upstream congestion with the size of 500 km × minutes in which multiple disturbances (e.g. 15 instances) emerge”*. A corresponding retrieval can be carried out by constructing a relation graph with two nodes, namely B and D. Their attributes are directly obtained from the description.

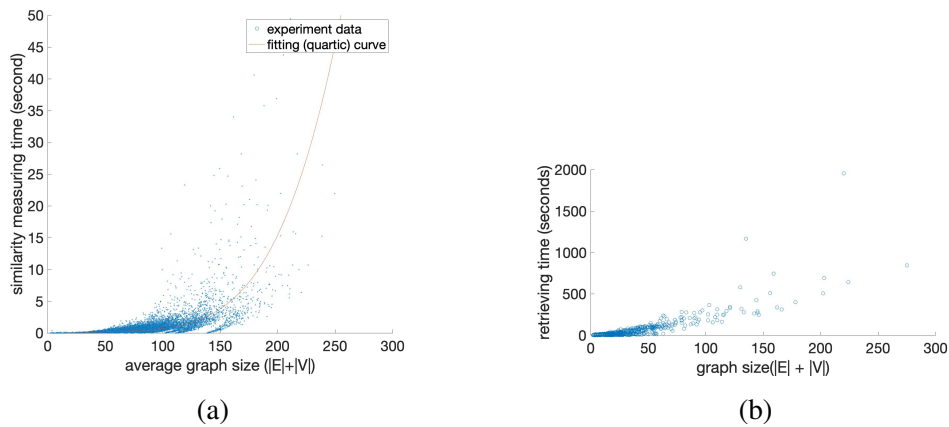


Figure 6.14: Computation time of measuring the similarity between two relation-graphs: (a) single pair measurement, (b) retrieval time from the whole data (of 778 congestion patterns).

6.5 Conclusion

This chapter presents a new method for pattern retrieval of highway traffic congestion based on image processing techniques and a graph matching approach. We demonstrate the efficacy and efficiency of the method on a large scale traffic database covering the entire Dutch freeway road network over several years.

From the image representations of congestion patterns, traffic-domain elements are extracted by applying image processing techniques. Specifically, those include disturbances (using the Watershed-based segmentation), bottlenecks (using gradient filtering based on the direction of characteristic waves), and homogeneous congestion (using texture-based analysis in this chapter). We propose a so-called relation graph as an abstract representation of the overall congestion patterns with their constituent components (bottlenecks, homogeneous congested regions and disturbances) so that their spatial relations are preserved. We formulate a parameterised matching function to measure the similarity between two relation graphs, reflecting different perspectives on observing patterns, namely component sizes, redundancy penalties, disturbance frequencies, individual completion and structural integrity.

The case study demonstrates the ability of the proposed method to successfully retrieve complex patterns with various retrieval cases. Most importantly, we show that combining domain knowledge with computer science techniques is highly effective. Using traffic-domain features help to make query outcomes transparent and easily explainable. Moreover, via the customisable parameters, modifications are made possible to improve a retrieval result according to users' points of view on similarity, i.e. what they are looking for. The relation-graph representation also offers a new opportunity for semantic retrieval, in which expected patterns are described intuitively using some well-known traffic phenomena. This successful development of a retrieval system for 2D congestion patterns essentially motivates various studies in, for example,

congestion analysis, traffic prediction. A broader and more thorough understanding of variations of a type of congestion is beneficial for evaluating mitigation strategies for the corresponding congested traffic. Furthermore, traffic states can be predicted at an abstract level, i.e. pattern level, which in essence is another way to look at traffic at a higher level.

There are open directions for future research to improve the proposed method. From a practical viewpoint, computation time is important. One promising approach is to start with a (rough) classification of the input pattern. Accordingly, the search space is effectively reduced from an entire dataset to the partition representing one single class of patterns. A different approach is a two-step approach that combines generic feature approaches and the proposed approach. Since Euclidean distances are fast to compute, similarities between patterns can be measured quickly based on the former features. Then, the proposed approach only needs to be applied to a reasonable number of top-ranked patterns thereof. Another future research direction is to incorporate more relevant characteristics to the relation-graph to refine retrieval outcomes.

Chapter 7

Conclusion

The research presented in this thesis is motivated by the objective of mining collected traffic data for relevant information, specifically patterns of congestion. A modular research approach has been proposed with various components which together formulate CoSI - Congestion Search EngIne. The realisation of CoSI includes both the construction of an underlying database and algorithm developments for corresponding data retrieval.

This chapter starts with key findings and the main conclusions are presented. Then, we discuss related implications for practice. And finally, we close the chapter with information on promising research directions resulting from contributions made in this thesis.

7.1 Main conclusions and Key findings

In this section, we summarise the key findings and main conclusions with respect to the four research questions stated in Chapter 1.

Research question 1. How can congestion patterns be collected periodically from traffic data from large-scale highway networks?

In Chapter 3, we proposed a framework for collecting spatio-temporal patterns of traffic congestion (which essentially represent the evolutions of vehicular speeds over space/route and time). It comprises of three stages, namely congestion detection, congestion clustering and pattern extraction. Accordingly, we proposed appropriate algorithms for different modules in these stages. In the first stage, a traffic network is cut into singular routes to benefit data interpolation, which results in finer estimations of traffic states thereof. Hence, traffic congestion can be detected more effectively. During the second stage, congested data points are associated (based on the underlying network topology) to formulate a so-called congestion graph, from which isolated clusters are determined to tailor further processing. Each congestion cluster is handled independently in the last stage. Complete propagating paths of congestion - represented as sub-graphs - are tracked and merged when possible. This results in a set of sub-graphs representing different complete congestion patterns. The analysis suggests that the overall complexity of the algorithms is most likely linear to the size of the network, or more precisely the size of the filtered data. The case study on the Dutch highway network indicates the feasibility of the proposed methodologies. Data for one entire day is processed in less than 40 minutes, which results in many congestion patterns.

Research question 2. What are the representative features of (highway) congestion patterns?

In Chapter 4, two types of feature schemes are analysed for the capability to discriminate congestion patterns. These are so-called generic (or point-based) and domain-specific (area-based). Even though both approaches consider traffic patterns as (intensity) images and apply image processing techniques, they are fundamentally different with respect to the semantic meanings of their derived features. On one hand, the former approach explores key points in (image) patterns which are related to either a blob, a corner or edges - i.e. object boundaries. These points, therefore, do not necessarily have a direct meaning related to traffic. On the other hand, the latter approach uses traffic-related elements as key features, namely the frequency of wide moving jam instances, the frequency of homogeneous congested regions, longitudinal and temporal extents of represented congestion. Both approaches use a histogram of defined features to formulate scalar feature vectors that act as representations of congestion patterns. The Euclidean distances on these vectors are then used as measurements for dissimilarities between congestion patterns. Since there is no ground-truth set of labels, the quantitative assessment tests how well both methods can partition the feature space they construct for the (large) database of traffic patterns. We argue that the more

crisp this separation is; the better the labelling has turned out. For this quantitative comparison, we trained a multinomial classifier that maps unseen patterns to the labels discovered by each of the two labelling approaches. The most important result is that the classifier using the area-based feature vector achieves the highest average levels of confidence in its decisions to classify patterns, implying a highly separable feature vector space. Not only does the combination of image processing (Watershed) and domain knowledge (traffic flow characteristics) lead to meaningful labels that can be automatically retrieved from large databases of data; this method also leads to more efficient separation of the resulting feature space.

Research question 3. How can domain-specific features, namely characteristics of traffic congestion which are well recognized in the literature, be learned automatically?

By answering the previous question, traffic domain-specific features are recognised as more beneficial to discrimination application of congestion patterns. Besides, as those patterns represent traffic in a two-dimensional manner, i.e. both spatially and temporally, they are seen as (intensity) images and that allow image-processing techniques be applied. This results in different methods for extracting traffic domain characteristics presented in Chapter 4 and Chapter 5.

With the focus on congested traffic, in Chapter 5, we first identified the spatial and temporal boundaries/extents of congestion by applying the Chan-Vese model. It is an active contour model which evolves a boundary from an initial state (usually set by relevant heuristics to achieve a fast convergence) to the state that separates congested and free-flow traffic. Subsequently, congested regions/segments can be effectively identified in spatio-temporal patterns of traffic speeds. This sufficiently paves a way for further extractions of congestion characteristics like spatial and temporal extents. In Chapter 4, the well-known Watershed operator is adopted to segment a pattern into different areas from which three main traffic-related elements are extracted—spatial and temporal scales, disturbances and homogeneously congested areas.

In Chapter 5, we proposed a framework to detect another dominant and well-observed phenomenon of traffic congestion which are the bottleneck activations. Speed discontinuities along the direction of characteristic waves provide an effective indicator for the activations of related bottlenecks. The evaluation suggests that the framework is capable of detecting multiple close-distance bottlenecks in singular patterns. In combination with the detected congested regions, relationships between bottlenecks can be identified, in particular, whether a bottleneck is primary or secondary. The method is automatic and, therefore, can be applied to a large amount of data over long periods and extract meaningful statistics for traffic managers.

Research question 4. How can similarities between congestion patterns be ranked using features that are previously learned?

Content-based pattern retrieval is a valuable application for databases of congestion patterns as it can benefit various analyses. For instance, by investigating the underlying traffic control strategies from similar congestion patterns, their efficiencies can

be compared and justified. In Chapter 6, we propose abstractly representing congestion patterns by so-called relational graphs, in which nodes encode semantic pieces within congestion patterns, and edges imply (temporal) relations between these pieces. Pattern similarities are then measured by the graph matching method with an iterative approach, i.e. similarities between a pair of nodes are aggregated with those from their descendent nodes. Additionally, some parameters are introduced to reflect potentially different perspectives when defining similar patterns. The case study qualitatively shows that the proposed approach is capable of locating similar patterns to a query pattern. The introduction of the relation graph also results in a new method for semantic retrieving, in which expected patterns can be manually constructed without being given an example image. In other words, one can use the node/link concepts in the relation graph to describe a pattern and use that as input for retrieval. Based on the emergence of this searching scheme, the exploration of databases of congestion patterns is made significantly more flexible and intuitive.

7.2 Implications for practice

The research in this thesis is dedicated to equip traffic management systems with an intelligent retrieval application for relevant congestion patterns from massive amounts of collected measurements. Several methods and algorithms have been designed and implemented. Various validations have also been conducted to test those methods. This section discusses the implications of these results for practice.

One of the main concerns in traffic is congestion, which essentially undermines the flow of vehicles through a road network. In Chapter 3, we proposed a framework and developed appropriate algorithms to detect all possible occurrences of congestion on a network over a certain period. This piece of research is valuable for practice in the way that different patterns of congestion can be automatically collected. They represent the dynamics of traffic in general and congestion in specific over both spatial and temporal dimensions. That is intuitively convenient for visualisation and observation. By having such data ready a priori, time and efforts for preprocessing are saved when the need for data analyses occurs.

Manually analysing thousands of movements of traffic congestion on a network is cumbersome if not impossible. The feature extraction techniques developed in Chapter 4 and Chapter 5 provide an automatic approach. Not only can they help to identify key features of congestion quickly, but also to process large amounts of data is made effectively possible. For example, frequent disturbances can be detected (Chapter 4) and measured for their upstream-propagating distances (with respect to wide moving jams). This information potentially provides a proxy in assessing the effectiveness of traffic control such as variable speed limits.

Bottlenecks are one of the dominant causes of traffic jams. Hence, detecting them can play a vital role in developing congestion mitigating strategies. In Chapter 5, a

comprehensive automatic detection framework for bottleneck activations on highways is presented. The feasibility of the proposed method enriches an ITS (intelligent transportation system) with an effective tool to identify various bottlenecks on a highway network. Consequences of bottleneck activations, i.e. the incurred congestion, are automatically determined by the method. Some examples are how many disturbances emerged, how homogeneous/heterogeneous is traffic during congestion, or how much delay is caused. Furthermore, the method is also applicable to dense networks where multiple bottlenecks might be involved in single instances of congestion. Relationships between these activations, like whether one triggers another, are identified. This provides information for traffic managers about potentially critical bottlenecks to focus the mitigation efforts on. More importantly, the method is automatic and can be applied to handle data for a longer time horizon. Hence, characteristics of bottlenecks can be derived and synthesized into informative statistics. This is significantly important for traffic management systems.

Furthermore, the success in extracting key characteristics for congestion patterns (including wide moving jam, homogeneity, bottleneck activation) helps generate a more abstract and intuitive layer for a traffic congestion database over raw collected data. And useful applications like data retrieval can be effectively built upon.

Pattern representation is significantly important based on which retrieval application can be conducted sufficiently. The so-called relation graph proposed in Chapter 6 is shown to be capable of providing intuitive representation for patterns of congestion. Nodes are substitutions for traffic phenomena and links imply their relations. This relation graph can be easily grasped by practitioners as they contain elements that are traffic-semantically understandable and well-observable. In addition, the heuristic definitions of costs incurred during the matching of such graphs consist of parameters that reflect different perspectives on pattern similarities. This gives flexibilities for users in describing their preferences on expected patterns from the retrieval system.

Finally, all the pieces of research in this thesis are greatly prepared for a realisation of a smart database system of congestion patterns, which is the so-called CoSI. Its design is discussed in Chapter 2. The developed algorithms are implemented and validated in appropriate case studies which ensure its validity. This implies that CoSI is highly ready for production as an advantageous component for any ITS system. By doing that, collected traffic data can be accessed through a more efficient information retrieval engine which we believe is significantly beneficial to many practice in the field of transportation.

7.3 Recommendations for future research

The research presented in this thesis is dedicated to improve the use of traffic data, in which, many methodologies are developed and key findings are shed light on. This

section discusses potential research directions that can benefit greatly from these outcomes of the thesis.

In Chapter 4, two feature schemes are investigated. The result suggests that both schemes have their unique strengths. While the domain-knowledge specific features are more effective in clustering congestion patterns, the generic features are more capable of identifying locally similar features, e.g. (image) textures. Combining these two approaches is worth analysing. For instance, a potential direction is to employ a hierarchical framework, in which the abstract features are at high levels and local features are at low levels. Accordingly, the structure of a dataset can be further partitioned which is valuable for many applications, e.g. classification, clustering and pattern retrieval.

Traffic prediction is certainly an important topic for both traffic managers and road users, in which, congestion is one of the most critical factors since it impairs greatly the quality of mobility. Being able to predict how congestion progresses is as important as when it will occur. The collected patterns as a result of research presented in Chapter 3 provide a rich input for understanding how congestion evolves both spatially and temporally. Although using traffic patterns for traffic prediction is not a new idea, we expect by combining it with pattern classification it can lead to fruitful outcomes. Specifically, instead of considering all available patterns of congestion occurred at a single place, only relevant patterns are selected (by an appropriate classifier) for predicting purposes. Hence, partial classification of congestion and quick retrieval of relevant patterns are promising directions for future research.

Bottlenecks are one of the main causes of traffic congestion. Understanding the interactions and/or dependencies between nearby bottlenecks is valuable for traffic prediction and/or control strategies. The automatic detection framework developed in Chapter 5 can serve as an important milestone for further studying bottlenecks on a higher level, such as nation-wide analysis. For example, an interesting direction is to identify and classify bottlenecks in a (whole/partial) traffic network based on their characteristics (or consequences), which can be measured from the related congestion patterns. In addition, a promising step could be to incorporate other sources of data such as network topology. The correlations (if any) between classes of bottlenecks and traffic control strategies are also worth assessing.

The retrieval application (see Chapter 6) on congestion patterns database is a valuable module for utilising collected traffic data. Despite most of the main required elements are studied and presented in this thesis, there is still opportunities for improvement. A promising direction is to incorporate user judgments (so-called relevant feedback), which is indicated by the (optional) loop in the overall CoSI framework design (see Chapter 2). The idea is that searching outcomes can be further revised based on user opinions. Furthermore, numerous increasing number of requests can potentially be expected once the system is deployed. If properly designed, these significantly relevant inputs can be collected for further interesting research like reinforcement learning,

which essentially learns from the relevant feedback for enhancing responses to future queries.

Traffic control is a vital part of intelligent traffic systems (ITS). Typically, the main purpose is to maintain high capacities of traffic network and/or to mitigate the impacts from congestion. The CoSI framework featured with (historical) spatio-temporal congestion patterns and pattern retrieval engine is a promising tool in the process of developing or analysing effective control strategies. For instance, sophisticated traffic control typically associates operations from multiple controllers at different locations. An interesting suggestion could be to identify relevant sets of relevant locations for associating their control strategies. Upon achievement, this can reduce the complexity of involving less-relevant controllers and simultaneously improve the efficiency of the whole control system. The methods developed for pattern collection paves the way for identifying clusters of related congestion patterns which is certainly beneficial for identifying locations of interest.

Another potential research direction is the analysis of different control strategies. Specifically, how a control algorithm affects traffic at the network level, i.e. taking into consideration the effects on upstream roads. These affects can be investigated by looking at large-scale patterns of congestion as they essentially trace congested traffic to all possible paths over time until the dissolution of congestion. Profiles of the derived patterns associated with each strategy can be created, which can include any important information such as congestion type/class, duration, propagating distance, delay, economic cost, etc. These can be used as a criterion for evaluating and/or comparing traffic control strategies.

Appendices

Appendix A

Effects of ASM parameters on the recognition of traffic disturbances

The ASM contains several parameters, which may affect the automated recognition of patterns (shapes of e.g. disturbances) in the (smoothed) speed or flow image. In this appendix we test the sensitivity of one representative gradient-based method, the Watershed method used in Chapter 4, to a selection of four parameters. We expect these four parameters to have the largest effect on the degree in which such gradient-based methods can recognize patterns in the smoothed images, because they govern the degree of smoothing and the directions in which the data are smoothed.

These parameters are σ and τ , which govern the width of the ASM kernels over space and time, respectively; and c_{free} and c_{cong} , the two wave speeds for freely flowing and congested traffic respectively, along which the smoothing kernels are skewed.

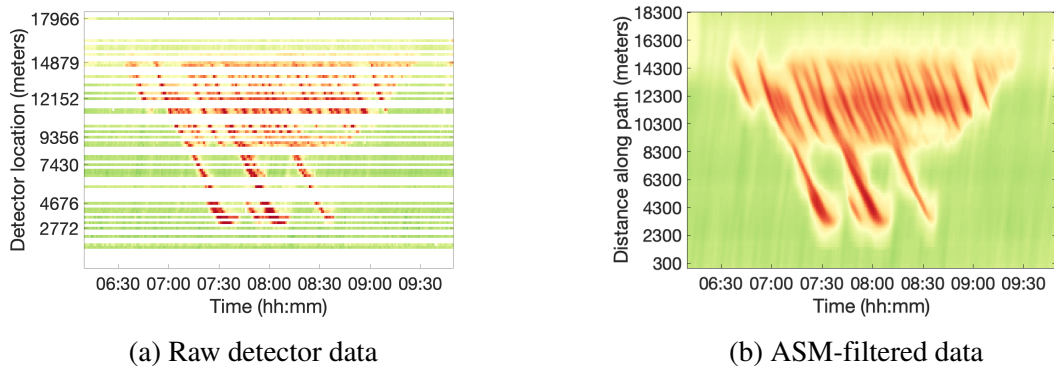


Figure A.1: A high frequency stop-and-go traffic congestion. The ASM parameters σ , τ , c_{cong} , c_{free} are 1000m, 60s, -18km/h, 80km/h, respectively.

To test the sensitivity of the Watershed method with respect to these parameters, we select a high frequency stop-and-go pattern for this analysis (see Fig. A.1). Visual examination of the pattern reveals 23 disturbances that we would also expect the Watershed method to be able to detect if perfect detection takes place.

A.1 Smoothing kernel broadness

To test the sensitivity with respect to σ , τ , we keep the values of c_{cong} and c_{free} constant. In particular, -18km/h and 80km/h are chosen for c_{cong} and c_{free} as recommended in the literature (Schreiter et al., 2010a; Treiber & Helbing, 2002). Two ranges of values for σ and τ are tested, namely 100 - 2000m and 30 - 180s, respectively. Various combinations of these values are applied to filter the raw data, and the resulting numbers of disturbances are compared with the expected value (i.e. 23). Fig. A.2 shows the obtained result. It is reasonable that as σ and τ increase, the number of disturbances decreases because ASM essentially uses a weighted smoothing kernel. When the widths of the kernel increase, nearby disturbances are strongly affected and (potentially) smoothed out, which results in blended regions (in corresponding spatiotemporal maps). (see Fig. A.2 for an example). The heat map of deviations suggests the most efficient range for τ is 60 - 90s, while the range for σ is more tolerable (as the deviations fluctuates less on rows than on columns in the heat map plot). From the plot, we recommend the range 1000 - 1500m for σ so that incurring errors are below 20 percent. Note that the heat map shows some groups of cells with different and low deviations, namely 0, 0.1 and 0.2. Hence, we choose 20 percent as the expected lower bound error for the recognition of disturbances.

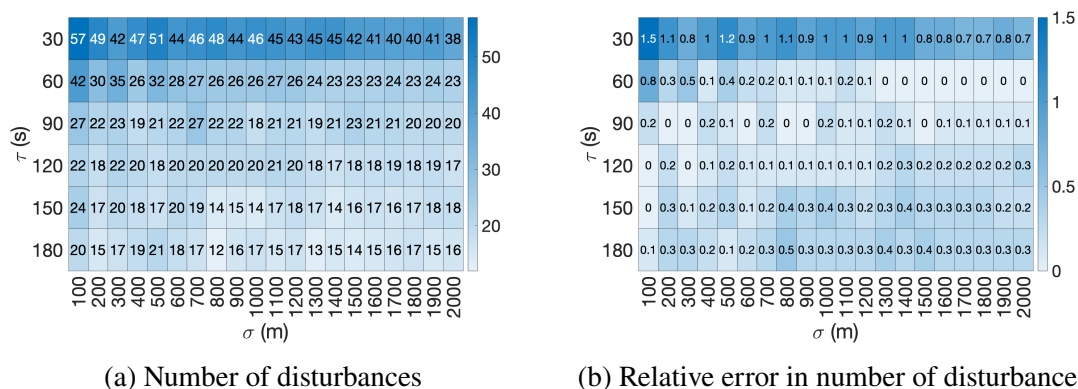


Figure A.2: The number of (recognised) disturbances with respect to different settings of σ and τ . The deviation is calculated as the proportion of the expected number.

A.2 Propagation speed

To test the sensitivity with respect to c_{cong} and c_{free} using the same approach as in the previous section, we now keep τ and σ constant at values recommended in literature (60s, 1000m) and evaluate the resulting numbers of disturbances when varying c_{cong} and c_{free} . Fig. A.4 shows the obtained result. Overall, increasing the absolute values of both propagation speeds reduces the number of (recognised) disturbances. Fig. A.4

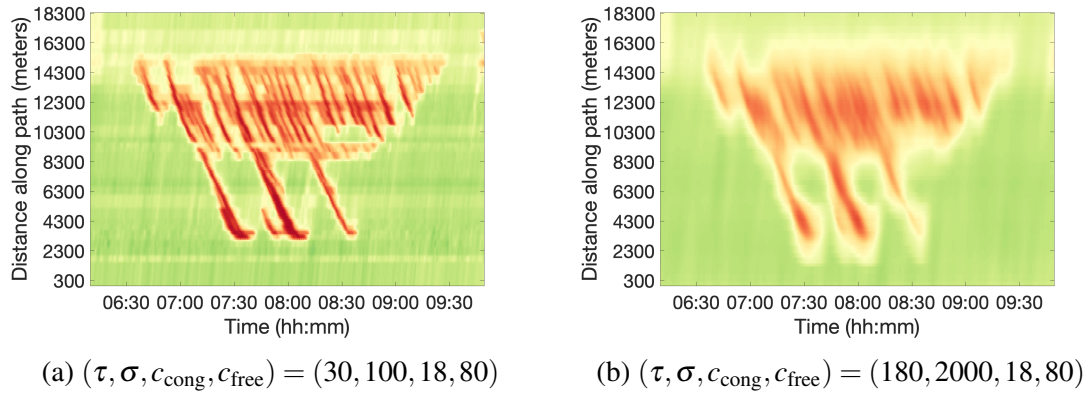
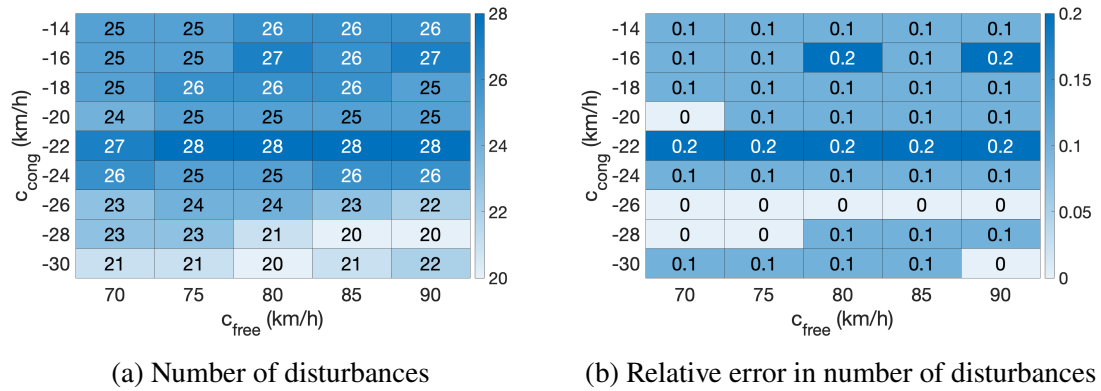


Figure A.3: The ASM-filtered patterns using two extreme settings

shows that the method is very robust with respect to variations in c_{free} , which makes sense: this parameter governs smoothing for non-congested areas.

Figure A.4: The number of (recognised) disturbances with respect to different settings of c_{cong} and c_{free} . The deviation is calculated as the proportion of the expected number.

As predicted by Treiber & Helbing (2002), the congested wave speed has the largest effect due to the so-called "egg-box" effect. If this parameter is chosen too large or too small, parallel moving jam waves are either cut in smaller pieces or "glued together", resulting in too many or too few recognized jam waves, respectively. Fig. A.4b illustrates that the Watershed method is fairly robust to variations in c_{cong} (within the given range) with a maximum relative error of again 20% in the number of recognized disturbances.

A possible automated method to minimize this error for any given circumstance is described in (Schreiter et al., 2010b), in which both wave speeds are estimated directly from raw data. Throughout this thesis, however, we have used the default values described in literature, that is, $c_{\text{free}} = 80$ km/h, and $c_{\text{cong}} = -18$ km/h.

Bibliography

- A12 highway, URL [https://en.wikipedia.org/wiki/A12_motorway_\(Netherlands\)](https://en.wikipedia.org/wiki/A12_motorway_(Netherlands)).
- A13 highway, URL [https://en.wikipedia.org/wiki/A13_motorway_\(Netherlands\)](https://en.wikipedia.org/wiki/A13_motorway_(Netherlands)).
- National datawarehouse of traffic information, URL <http://www.ndw.nu/en/>.
- Ackoff, R. L. (1989) From data to wisdom, *Journal of applied systems analysis*, 16(1), pp. 3–9.
- Bai, Y., Z. Wu, S. Sun, C. Wang (2011) Automatic identification algorithm for freeway bottleneck, in: *Transportation, Mechanical, and Electrical Engineering (TMEE), 2011 International Conference on*, IEEE, pp. 1857–1860.
- Ban, X., L. Chu, H. Benouar (2007) Bottleneck identification and calibration for corridor management planning, *Transportation Research Record: Journal of the Transportation Research Board*, (1999), pp. 40–53.
- Banks, J. H. (1990) Flow processes at a freeway bottleneck, *Transportation Research Record*, (1287).
- Banks, J. H. (1991) Two-capacity phenomenon at freeway bottlenecks: a basis for ramp metering, *Transportation Research Record*, 1320, pp. 64–69.
- Bay, H., A. Ess, T. Tuytelaars, L. Van Gool (2008) Speeded-up robust features (surf), *Computer vision and image understanding*, 110(3), pp. 346–359.
- Bertini, R. L., Z. Horowitz (2008) Diagnosing a freeway bottleneck in portland, oregon (usa) using archived sensor data, in: *Traffic and Transportation Studies*, pp. 815–827.
- Beucher, S., F. Meyer (1992) The morphological approach to segmentation: the watershed transformation, *Optical Engineering-New York-Marcel Dekker Incorporated-*, 34, pp. 433–433.
- Blondel, V. D., A. Gajardo, M. Heymans, P. Senellart, P. Van Dooren (2004) A measure of similarity between graph vertices: Applications to synonym extraction and web searching, *SIAM review*, 46(4), pp. 647–666.

- Böhning, D. (1992) Multinomial logistic regression algorithm, *Annals of the Institute of Statistical Mathematics*, 44(1), pp. 197–200.
- Bunke, H. (1997) On a relation between graph edit distance and maximum common subgraph, *Pattern Recognition Letters*, 18(8), pp. 689–694.
- Calvert, S., H. Taale, M. Snelder, S. Hoogendoorn (2018) Improving traffic management through consideration of uncertainty and stochastics in traffic flow, *Case Studies on Transport Policy*, 6(1), pp. 81–93.
- Calvert, S. C., T. A. Van Den Broek, M. van Noort (2011) Modelling cooperative driving in congestion shockwaves on a freeway network, in: *2011 14th International IEEE Conference on Intelligent Transportation Systems (ITSC)*, IEEE, pp. 614–619.
- Canny, J. (1986) A computational approach to edge detection, *IEEE Transactions on pattern analysis and machine intelligence*, (6), pp. 679–698.
- Cassidy, M. J., J. R. Windover (1995) Methodology for assessing dynamics of freeway traffic flow.
- Celikoglu, H. B. (2013) An approach to dynamic classification of traffic flow patterns, *Computer-Aided Civil and Infrastructure Engineering*, 28(4), pp. 273–288.
- Celikoglu, H. B., M. A. Silgu (2016) Extension of traffic flow pattern dynamic classification by a macroscopic model using multivariate clustering, *Transportation Science*, 50(3), pp. 966–981.
- Chan, T. F., L. A. Vese (2001) Active contours without edges, *IEEE Transactions on image processing*, 10(2), pp. 266–277.
- Chen, C., A. Skabardonis, P. Varaiya (2004) Systematic identification of freeway bottlenecks, *Transportation Research Record: Journal of the Transportation Research Board*, (1867), pp. 46–52.
- Chen, H., H. A. Rakha (2017) Automatic freeway bottleneck identification and visualization using image processing techniques, Tech. rep.
- Chui, C. K. (2016) *An introduction to wavelets*, Elsevier.
- Conte, D., P. Foggia, C. Sansone, M. Vento (2004) Thirty years of graph matching in pattern recognition, *International journal of pattern recognition and artificial intelligence*, 18(03), pp. 265–298.
- Cootes, T. F., C. J. Taylor, D. H. Cooper, J. Graham (1995) Active shape models-their training and application, *Computer vision and image understanding*, 61(1), pp. 38–59.
- Cortes, C., V. Vapnik (1995) Support-vector networks, *Machine learning*, 20(3), pp. 273–297.

- Cover, T., P. Hart (1967) Nearest neighbor pattern classification, *IEEE transactions on information theory*, 13(1), pp. 21–27.
- Das, S., D. Levinson (2004) Queuing and statistical analysis of freeway bottleneck formation, *Journal of transportation engineering*, 130(6), pp. 787–795.
- Datta, R., D. Joshi, J. Li, J. Z. Wang (2008) Image retrieval: Ideas, influences, and trends of the new age, *ACM Computing Surveys (Csur)*, 40(2), pp. 1–60.
- Depaire, B., G. Wets, K. Vanhoof (2008) Traffic accident segmentation by means of latent class clustering, *Accident Analysis & Prevention*, 40(4), pp. 1257–1266.
- Elhenawy, M., H. A. Rakha, H. Chen (2013) An automated statistically-principled bottleneck identification algorithm (asbia), in: *Intelligent Transportation Systems- (ITSC), 2013 16th International IEEE Conference on*, IEEE, pp. 1846–1851.
- Emmert-Streib, F., M. Dehmer, Y. Shi (2016) Fifty years of graph matching, network alignment and network comparison, *Information sciences*, 346, pp. 180–197.
- Fahad, A., N. Alshatri, Z. Tari, A. Alamri, I. Khalil, A. Y. Zomaya, S. Fofou, A. Bouras (2014) A survey of clustering algorithms for big data: Taxonomy and empirical analysis, *IEEE transactions on emerging topics in computing*, 2(3), pp. 267–279.
- Falocchio, J. C., H. S. Levinson (2015) The costs and other consequences of traffic congestion, in: *Road Traffic Congestion: A Concise Guide*, Springer, pp. 159–182.
- Foggia, P., G. Percannella, M. Vento (2014) Graph matching and learning in pattern recognition in the last 10 years, *International Journal of Pattern Recognition and Artificial Intelligence*, 28(01), p. 1450001.
- Gao, J., G. Andrew, M. Johnson, K. Toutanova (2007) A comparative study of parameter estimation methods for statistical natural language processing.
- Gao, X., B. Xiao, D. Tao, X. Li (2010) A survey of graph edit distance, *Pattern Analysis and applications*, 13(1), pp. 113–129.
- Gärtner, T., P. Flach, S. Wrobel (2003) On graph kernels: Hardness results and efficient alternatives, in: *Learning theory and kernel machines*, Springer, pp. 129–143.
- Hale, D., R. Jagannathan, M. Xyntarakis, P. Su, X. Jiang, J. Ma, J. Hu, C. Krause, et al. (2016) Traffic bottlenecks: identification and solutions, Tech. rep., United States. Federal Highway Administration. Office of Operations Research
- Hall, F. L., K. Agyemang-Duah (1991) Freeway capacity drop and the definition of capacity, *Transportation research record*, (1320).

- Hao, P., C. Wang, G. Wu, K. Boriboonsomsin, M. Barth (2017) Evaluating the environmental impact of traffic congestion based on sparse mobile crowd-sourced data, in: *2017 IEEE Conference on Technologies for Sustainability (SusTech)*, IEEE, pp. 1–6.
- Haralick, R. M., K. Shanmugam, I. H. Dinstein (1973) Textural features for image classification, *IEEE Transactions on systems, man, and cybernetics*, (6), pp. 610–621.
- Haralick, R. M., S. R. Sternberg, X. Zhuang (1987) Image analysis using mathematical morphology, *IEEE transactions on pattern analysis and machine intelligence*, (4), pp. 532–550.
- Harris, C., M. Stephens (1988) A combined corner and edge detector., in: *Alvey vision conference*, vol. 15, Citeseer, pp. 10–5244.
- Helbing, D., A. Hennecke, M. Treiber (1999) Phase diagram of traffic states in the presence of inhomogeneities, *Physical Review Letters*, 82(21), p. 4360.
- Helbing, D., M. Treiber, A. Kesting, M. Schönhof (2009) Theoretical vs. empirical classification and prediction of congested traffic states, *The European Physical Journal B-Condensed Matter and Complex Systems*, 69(4), pp. 583–598.
- Huang, F.-L., C.-J. Hsieh, K.-W. Chang, C.-J. Lin (2010) Iterative scaling and coordinate descent methods for maximum entropy models, *Journal of Machine Learning Research*, 11(Feb), pp. 815–848.
- Hymel, K. (2009) Does traffic congestion reduce employment growth?, *Journal of Urban Economics*, 65(2), pp. 127–135.
- Jain, A. K. (2010) Data clustering: 50 years beyond k-means, *Pattern recognition letters*, 31(8), pp. 651–666.
- Jin, J., P. Rafferty (2017) Does congestion negatively affect income growth and employment growth? empirical evidence from us metropolitan regions, *Transport Policy*, 55, pp. 1–8.
- Jin, P., S. Parker, J. Fang, B. Ran, C. M. Walton (2012) Freeway recurrent bottleneck identification algorithms considering detector data quality issues, *Journal of Transportation Engineering*, 138(10), pp. 1205–1214.
- Kadir, T., M. Brady (2001) Saliency, scale and image description, *International Journal of Computer Vision*, 45(2), pp. 83–105.
- Kerner, B. S. (2002) Empirical macroscopic features of spatial-temporal traffic patterns at highway bottlenecks, *Physical Review E*, 65(4), p. 046138.
- Kerner, B. S., H. Rehborn (1996) Experimental features and characteristics of traffic jams, *Physical Review E*, 53(2), p. R1297.

- Kerner, B. S., H. Rehborn, M. Aleksic, A. Haug (2004) Recognition and tracking of spatial-temporal congested traffic patterns on freeways, *Transportation Research Part C: Emerging Technologies*, 12(5), pp. 369–400.
- KiM, Transport and mobility, https://en.wikipedia.org/wiki/Road_transport_in_the_Netherlands.
- Kim, J., H. S. Mahmassani (2015a) Spatial and temporal characterization of travel patterns in a traffic network using vehicle trajectories, *Transportation Research Part C*, (59), pp. 375–390.
- Kim, J., H. S. Mahmassani (2015b) Trajectory clustering for discovering spatial traffic flow patterns in road networks, Tech. rep.
- Krishnakumari, P., T. Nguyen, L. Heydenrijk-Ottens, H. L. Vu, H. van Lint (2017) Traffic congestion pattern classification using multiclass active shape models, *Transportation Research Record: Journal of the Transportation Research Board*, (2645), pp. 94–103.
- Kuhn, H. W. (1955) The hungarian method for the assignment problem, *Naval research logistics quarterly*, 2(1-2), pp. 83–97.
- Lee, H., H.-W. Lee, D. Kim (2000) Phase diagram of congested traffic flow: An empirical study, *Physical Review E*, 62(4), p. 4737.
- Lopez, C., L. Leclercq, P. Krishnakumari, N. Chiabaut, H. van Lint (2017) Revealing the day-to-day regularity of urban congestion patterns with 3d speed maps, *Scientific Reports*, 7(1), pp. 1–11.
- Lowe, D. G. (1999) Object recognition from local scale-invariant features, in: *Computer vision, 1999. The proceedings of the seventh IEEE international conference on*, vol. 2, Ieee, pp. 1150–1157.
- Ministry of Infrastructure and Water Management, The netherlands network shape files, https://www.rijkswaterstaat.nl/apps/geoservices/geodata/dmc/nwb-wegen/geogegevens/shapefile/Nederland_totaal/.
- Netherlands Transport, Road transport in the Netherlands, https://en.wikipedia.org/wiki/Road_transport_in_the_Netherlands.
- Nguyen, H. N., P. Krishnakumari, H. L. Vu, H. van Lint (2016) Traffic congestion pattern classification using multi-class svm, in: *Intelligent Transportation Systems (ITSC), 2016 IEEE 19th International Conference on*, IEEE, pp. 1059–1064.
- Nguyen, T. T., S. C. Calvert, H. L. Vu, H. van Lint (2021) An automated detection framework for multiple highway bottleneck activations, *IEEE Transactions on Intelligent Transportation Systems*.

- Nguyen, T. T., P. Krishnakumari, S. C. Calvert, H. L. Vu, H. Van Lint (2019) Feature extraction and clustering analysis of highway congestion, *Transportation Research Part C: Emerging Technologies*, 100, pp. 238–258.
- Osher, S., J. A. Sethian (1988) Fronts propagating with curvature-dependent speed: algorithms based on hamilton-jacobi formulations, *Journal of computational physics*, 79(1), pp. 12–49.
- Palmer, J., R. L. Bertini, H. Rehborn, J. Wiecek, R. J. Fernández-Moctezuma (2009) Comparing a bottleneck identification tool with the congested traffic pattern recognition system asda/foto using archived freeway data from portland, oregon, in: *Proc., 16th World Congress on ITS*.
- Rambabu, C., I. Chakrabarti (2008) An efficient hillclimbing-based watershed algorithm and its prototype hardware architecture, *Journal of Signal Processing Systems*, 52(3), pp. 281–295.
- Ray, S., R. H. Turi (1999) Determination of number of clusters in k-means clustering and application in colour image segmentation, in: *Proceedings of the 4th international conference on advances in pattern recognition and digital techniques*, Calcutta, India, pp. 137–143.
- Rendón, E., I. Abundez, A. Arizmendi, E. M. Quiroz (2011) Internal versus external cluster validation indexes, *International Journal of computers and communications*, 5(1), pp. 27–34.
- Riesen, K. (2015) Structural pattern recognition with graph edit distance, in: *Advances in computer vision and pattern recognition*, Springer.
- Rowley, J. (2007) The wisdom hierarchy: representations of the dikw hierarchy, *Journal of information science*, 33(2), pp. 163–180.
- Schönhof, M., D. Helbing (2007) Empirical features of congested traffic states and their implications for traffic modeling, *Transportation Science*, 41(2), pp. 135–166.
- Schreiter, T., H. van Lint, M. Treiber, S. Hoogendoorn (2010a) Two fast implementations of the adaptive smoothing method used in highway traffic state estimation, in: *Intelligent Transportation Systems (ITSC), 2010 13th International IEEE Conference on*, IEEE, pp. 1202–1208.
- Schreiter, T., H. Van Lint, Y. Yuan, S. Hoogendoorn (2010b) Propagation wave speed estimation of freeway traffic with image processing tools, Tech. rep.
- Sobel, I. (1990) An isotropic 3×3 image gradient operator, *Machine vision for three-dimensional scenes*, pp. 376–379.
- Soriguera, F., F. Robusté (2011) Estimation of traffic stream space mean speed from time aggregations of double loop detector data, *Transportation research part C: emerging technologies*, 19(1), pp. 115–129.

- Spiliopoulou, A., M. Kontorinaki, M. Papageorgiou, P. Kopelias (2014) Macroscopic traffic flow model validation at congested freeway off-ramp areas, *Transportation Research Part C: Emerging Technologies*, 41, pp. 18–29.
- Systematics, C. (2004) Traffic congestion and reliability: Linking solutions to problems, Tech. rep., United States. Federal Highway Administration.
- Tarjan, R. (1972) Depth-first search and linear graph algorithms, *SIAM journal on computing*, 1(2), pp. 146–160.
- Treiber, M., D. Helbing (2002) Reconstructing the spatio-temporal traffic dynamics from stationary detector data, *Cooperative Transportation Dynamics*, 1(3), pp. 3–1.
- van de Weg, G. S., H. L. Vu, A. Hegyi, S. P. Hoogendoorn (2018) A hierarchical control framework for coordination of intersection signal timings in all traffic regimes, *IEEE Transactions on Intelligent Transportation Systems*, 20(5), pp. 1815–1827.
- Van Lint, J., S. Hoogendoorn, H. J. van Zuylen (2005) Accurate freeway travel time prediction with state-space neural networks under missing data, *Transportation Research Part C: Emerging Technologies*, 13(5-6), pp. 347–369.
- Van Lint, J., S. P. Hoogendoorn (2010) A robust and efficient method for fusing heterogeneous data from traffic sensors on freeways, *Computer-Aided Civil and Infrastructure Engineering*, 25(8), pp. 596–612.
- Van Lint, J., H. J. Van Zuylen, H. Tu (2008) Travel time unreliability on freeways: Why measures based on variance tell only half the story, *Transportation Research Part A: Policy and Practice*, 42(1), pp. 258–277.
- Vermijs, R., H. Schuurman (1994) Evaluating capacity of freeway weaving sections and on-ramps using the microscopic simulation model fosim, in: *Proceedings of the second international symposium on highway capacity*, vol. 2.
- Vlahogianni, E. I., M. G. Karlaftis, J. C. Golias (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.
- Wang, Y., M. Papageorgiou, A. Messmer (2006) Renaissance—a unified macroscopic model-based approach to real-time freeway network traffic surveillance, *Transportation Research Part C: Emerging Technologies*, 14(3), pp. 190–212.
- Wieczorek, J., R. Fernández-Moctezuma, R. Bertini (2010) Techniques for validating an automatic bottleneck detection tool using archived freeway sensor data, *Transportation Research Record: Journal of the Transportation Research Board*, (2160), pp. 87–95.
- Witten, I. H., E. Frank, M. A. Hall, C. J. Pal (2016) *Data Mining: Practical machine learning tools and techniques*, Morgan Kaufmann.

- Xu, D., Y. Tian (2015) A comprehensive survey of clustering algorithms, *Annals of Data Science*, 2(2), pp. 165–193.
- Xu, R., D. Wunsch (2005) Survey of clustering algorithms, *IEEE Transactions on neural networks*, 16(3), pp. 645–678.
- Yang, Y., M. Li, J. Yu, F. He (2020) Expressway bottleneck pattern identification using traffic big data—the case of ring roads in beijing, china, *Journal of Intelligent Transportation Systems*, 24(1), pp. 54–67.
- Yildirimoglu, M., N. Geroliminis (2013) Experienced travel time prediction for congested freeways, *Transportation Research Part B: Methodological*, 53, pp. 45–63.
- Yu, H.-F., F.-L. Huang, C.-J. Lin (2011) Dual coordinate descent methods for logistic regression and maximum entropy models, *Machine Learning*, 85(1-2), pp. 41–75.
- Zager, L. A., G. C. Verghese (2008) Graph similarity scoring and matching, *Applied mathematics letters*, 21(1), pp. 86–94.
- Zhang, J.-b., G.-h. Song, L. Yu, J.-f. Guo, H.-y. Lu (2018) Identification and characteristics analysis of bottlenecks on urban expressways based on floating car data, *Journal of Central South University*, 25(8), pp. 2014–2024.
- Zhang, L., D. Levinson (2004) Some properties of flows at freeway bottlenecks, *Transportation Research Record: Journal of the Transportation Research Board*, (1883), pp. 122–131.
- Zhang, L., D. Levinson (2010) Ramp metering and freeway bottleneck capacity, *Transportation Research Part A: Policy and Practice*, 44(4), pp. 218–235.
- Zheng, N., R. A. Waraich, K. W. Axhausen, N. Geroliminis (2012) A dynamic cordon pricing scheme combining the macroscopic fundamental diagram and an agent-based traffic model, *Transportation Research Part A: Policy and Practice*, 46(8), pp. 1291–1303.
- Zhou, W., H. Li, Q. Tian (2017) Recent advance in content-based image retrieval: A literature survey, *arXiv preprint arXiv:1706.06064*.

Summary

Traffic congestion occurs daily, which can have negative effects on not only the quality of mobility, but also other important aspects of life like economic growth, health and environment. Both understanding and efficiently managing traffic are therefore crucially important tasks. Vast amounts of data are collected daily to gain insights into the dynamics of traffic. However, these data are typically stored in the form of raw measurements, that might hamper their potential benefits to both researchers and practitioners. A more informative and compact way to store traffic data is in the form of spatio-temporal maps, which have been shown to have advantage in intuitively observing traffic states. However, collecting, managing and retrieving such 2D patterns of congested traffic on large networks are challenging tasks. Accordingly, this dissertation is dedicated to developing methodologies and tools to advance the utilisation of traffic data, in particular, congestion patterns.

A conceptual framework for an intelligent search engine for congestion patterns (so-called CoSI) is designed. It covers the entire requirements necessary to develop such a system, ranging from processing raw data to searching through the resulting database of congestion patterns. Overall, the framework consists of two parts: database construction and search application (or so-called pattern retrieval). Their designs and relations are comprehensively presented in this research. The database construction is responsible for preparing a database of patterns of congested traffic which is carefully designed for the conveniences of a search application. Its conceptual design consists of three layers (or phases): pattern collection, feature extraction and pattern annotation. Regarding the search application, several possibilities for retrieving patterns are identified in association with the aforementioned steps of constructing the underlying database. These pieces of research are summarised as follows.

Congestion database construction

For the lowest layer in the database construction, this dissertation examines how to harvest spatio-temporal patterns of congestion from local measurements on a highway traffic network. This entails acknowledging and addressing several challenges. In a road network, congestion can propagate from an initial location to upstream intersections and junctions. From there, spill-backs to different roads are possible. In addition,

scalability is an important aspect that needs to be considered. As the size of the underlying network becomes bigger, the need for a more efficient methodology occurs. Taking these two requirements into consideration, a pattern collection framework is proposed which includes several components. First, a heuristic partition method is proposed to divide a traffic network into individual paths, from which traffic states are estimated in a finer resolution. Congestion is detected based on the filtered data, then followed by the construction of a three-dimensional graph associating related regions of congestion at the network link level. This level switching significantly simplifies the problem in terms of data size. A depth-first-search based method is developed to traverse this graph in search for patterns of congestion. The whole framework is applied to loop-detector data on the Dutch national highway network. Different clusters of congestion are identified, from which many congestion patterns are extracted, which provide a more intuitively observable representation of traffic congestion. The resulting processing time suggests the feasibility of applying the framework to large-scale traffic network.

The regular occurrence of traffic congestion can lead to a vast number of congestion patterns accumulated over time. This poses a challenge for effectively retrieving relevant congestion patterns, for instance, to locate a certain type of patterns that represent a similar dynamic of congestion. The top two layers of the database construction, feature extraction and pattern annotation, are designed to address this challenge. The former identifies salient traits, i.e. so-called features, in a congestion pattern that separate it from other patterns. The latter builds a classifier based on these features to annotate patterns. In particular, this dissertation investigates two fundamentally different feature schemes, which both use image representation for congestion patterns. The first one, so-called point-based approach, uses a histogram of generic key points in (pattern) images as feature vectors. The second one, so-called area-based approach, applies image segmentation to extract features at a higher level of abstraction by associating relevant traffic domain knowledge to image regions. Clustering analysis and classification are used to evaluate these approaches. Both the qualitative and quantitative results indicate the dominance of the area-based features in dividing congestion patterns into more homogeneous groups. In addition, this directly results in interpretable features and yields a much smaller feature space. On the other hand, the point-based approach is found capable of recognizing more detailed and subtle differences between clusters. Therefore, a combination of these feature approaches possibly derives finer clusters of congestion patterns.

The above study on the two feature schemes has demonstrated the advantages of using domain-knowledge to formulate features for congestion patterns. This finding motivates the study of traffic bottleneck activations, which empirical studies commonly observe. Detecting bottlenecks and characterising their traits from patterns of congestion can provide relevant inputs to investigate mitigating strategies. This dissertation develops an automated detection framework for (stationary) highway bottleneck activations. It employs two fundamental observations as primary indicators for activation

of bottlenecks, namely the temporal decrease of traffic speeds at a fixed location or in its vicinity and the prominence of speed drops along the direction of characteristic waves. The results suggest that the combination of raw and ASM-filtered data lead to an efficient identification method for bottleneck activation locations. In particular, raw data preserves speed discontinuities while smoothed data support the determination of precise activated locations. Furthermore, to characterise the consequence of a bottleneck activation, the spatio-temporal fronts of the upstream congestion are identified by applying image processing methods. Subsequently, primary and secondary bottlenecks are identified and provide insights into the interaction between nearby bottlenecks on a road network. This information is valuable for traffic managers and traffic controllers. In addition, the developed framework is applied to one-year of traffic data on a selected corridor of the Dutch highway system. Several bottlenecks have been identified, and their relationships, i.e. the order of activations, have also been determined.

Pattern retrieval application

The first part of the CoSI framework, including pattern collection, feature extraction and pattern annotation, is covered in the aforementioned research. The second part - pattern retrieval - relates to the application of searching through a (congestion pattern) database for relevant patterns. The conceptual design of CoSI identifies three possibilities for such an application: contextual, keyword-based and similarity-based. The first and second options are directly derived from the pattern collection and pattern annotation, respectively. Whilst, the last option searches for patterns by providing one example pattern. This is the objective of the final piece of research in this dissertation.

The feature extraction study develops methods for identifying domain-specific features in each congestion pattern. Based on these elements, this dissertation proposes a content-based approach, which includes relation-graph formulation and similarity measurement. The former aims at preserving the spatio-temporal relations between different regions of congestion patterns, each of which represents a traffic-domain feature. The latter develops a methodology with heuristic and parametrised matching functions for comparing relation-graphs representing congestion patterns. For validation, a collection of hundreds of patterns are used. Each time, patterns are ranked based on their similarities to a pre-selected pattern. The ranking results suggest the feasibility of the proposed methods in identifying similar patterns, i.e. assigning them to high ranks. The research further analyses the meanings and practical use of the parameters used in the graph matching function. In essence, each of these parameters reflects a different aspect of congestion patterns that can alter similarity scores. This enables informed users to interact with the retrieval system to achieve their expected results.

To conclude, this dissertation develops multiple frameworks and methodologies that are particularly valuable for advancing the utilisation of collected traffic data, from individual measurements to congestion patterns. As data have continuously been collected, the contribution of the studies should also become more significant. First, con-

gestion can be harvested automatically in forms of spatio-temporal patterns (*pattern collection* module). Second, many of its characteristics are also derived automatically (*feature extraction* module). Last but not least, relevant patterns in the resulted database can be retrieved intuitively and semantically with the relation-graph (*pattern retrieval* module).

About the Author

Tin T. Nguyen was born on August 18, 1988, in Hai Son, Quang Tri Province, which is in the middle of Vietnam. He finished secondary school in his home town before moving to a close-by city, Hue - the former imperial capital of Vietnam, to study high school with a special focus on mathematics. In 2006, he went to the South for his bachelor at one of the best universities in Vietnam - Ho-Chi-Minh City University of Technology. In 2011, he was awarded his bachelor degree with a major in Computer Engineering.



After his graduation, Tin worked as a research assistant in the same faculty for a few years. He was involved in several projects, in which his main roles were to develop algorithms such as a pedometer or an image-based barcode decoder. In 2014, he completed his study for a Master of Engineering degree in Computer Science. His thesis was titled "Formal verification of asynchronous circuit designs".

In 2016, Tin started his PhD education at Dittlab (Data Analytics and Traffic Simulation laboratory), at the Department of Transport & Planning, Delft University of Technology. His research was funded by NDW (National Data Warehouse) - a national organization in the Netherlands that manages a massive amount of traffic data. His PhD research focuses on patterns of traffic congestion on highway networks. He applies his knowledge in image processing, computer science, and machine learning to develop algorithms and methodologies for collecting congestion patterns, extracting salient features, annotating labels and retrieving similar patterns.

Journal Articles

1. Krishnakumari, P., T. Nguyen, L. Heydenrijk-Ottens, H. L. Vu, H. van Lint (2017) Traffic congestion pattern classification using multiclass active shape models, *Transportation Research Record: Journal of the Transportation Research Board*, (2645), pp. 94–103.
2. Nguyen, T. T., P. Krishnakumari, S. C. Calvert, H. L. Vu, H. Van Lint (2019) Feature extraction and clustering analysis of highway congestion, *Transportation Research Part C: Emerging Technologies*, 100, pp. 238–258.
3. Van Lint, H., Nguyen, T.T., Krishnakumari, P., Calvert, S.C., Schuurman, H. and Schreuder, M., (2020) Estimating the Safety Effects of Congestion Warning Systems using Carriageway Aggregate Data, *Transportation Research Record*, 2674(11), pp.278-288.
4. Nguyen, T. T., S. C. Calvert, H. L. Vu, H. Van Lint (2021) An automatic detection framework for multiple highway bottleneck activations, *IEEE Transactions on Intelligent Transportation Systems*.

Peer-reviewed Conference Contribution

1. Nguyen, T.T., Calvert, S.C. and van Lint, J.W.C., (2017) Hybrid traffic state estimation and prediction using pattern recognition. hEART 2017.

TRAIL Thesis Series

The following list contains the most recent dissertations in the TRAIL Thesis Series. For a complete overview of more than 275 titles see the TRAIL website: www.rsTRAIL.nl.

The TRAIL Thesis Series is a series of the Netherlands TRAIL Research School on transport, infrastructure and logistics.

Nguyen, T.T., *Highway Traffic Congestion Patterns: Feature Extraction and Pattern Retrieval*, T2021/22, July 2021, TRAIL Thesis Series, the Netherlands

Pudāne, B., *Time Use and Travel Behaviour with Automated Vehicles*, T2021/21, July 2021, TRAIL Thesis Series, the Netherlands

Gent, P. van, *Your Car Knows Best*, T2021/20, July 2021, TRAIL Thesis Series, the Netherlands

Wang, Y., *Modeling Human Spatial Behavior through Big Mobility Data*, T2021/19, June 2021, TRAIL Thesis Series, the Netherlands

Coevering, P. van de, *The Interplay between Land Use, Travel Behaviour and Attitudes: a quest for causality*, T2021/18, June 2021, TRAIL Thesis Series, the Netherlands

Landman, R., *Operational Control Solutions for Traffic Management on a Network Level*, T2021/17, June 2021, TRAIL Thesis Series, the Netherlands

Zomer, L.-B., *Unravelling Urban Wayfinding: Studies on the development of spatial knowledge, activity patterns, and route dynamics of cyclists*, T2021/16, May 2021, TRAIL Thesis Series, the Netherlands

Núñez Velasco, J.P., *Should I Stop or Should I Cross? Interactions between vulnerable road users and automated vehicles*, T2021/15, May 2021, TRAIL Thesis Series, the Netherlands

Duivenvoorden, K., *Speed Up to Safe Interactions: The effects of intersection design and road users' behaviour on the interaction between cyclists and car drivers*, T2021/14, April 2021, TRAIL Thesis Series, the Netherlands

Nagalur Subraveti, H.H.S., *Lane-Specific Traffic Flow Control*, T2021/13, March 2021, TRAIL Thesis Series, the Netherlands

Beirigo, B.A., *Dynamic Fleet Management for Autonomous Vehicles: Learning- and optimization-based strategies*, T2021/12, March 2021, TRAIL Thesis Series, the Netherlands

- Zhang, B., *Taking Back the Wheel: Transition of control from automated cars and trucks to manual driving*, T2021/11, February 2021, TRAIL Thesis Series, the Netherlands
- Boelhouwer, A., *Exploring, Developing and Evaluating In-Car HMI to Support Appropriate use of Automated Cars*, T2021/10, January 2021, TRAIL Thesis Series, the Netherlands
- Li, X., *Development of an Integrity Analytical Model to Predict the Wet Collapse Pressure of Flexible Risers*, T2021/9, February 2021, TRAIL Thesis Series, the Netherlands
- Li, Z., *Surface Crack Growth in Metallic Pipes Reinforced with Composite Repair System*, T2021/8, January 2021, TRAIL Thesis Series, the Netherlands
- Gavriilidou, A., *Cyclists in Motion: From data collection to behavioural models*, T2021/7, February 2021, TRAIL Thesis Series, the Netherlands
- Methorst, R., *Exploring the Pedestrians Realm: An overview of insights needed for developing a generative system approach to walkability*, T2021/6, February 2021, TRAIL Thesis Series, the Netherlands
- Walker, F., *To Trust or Not to Trust? Assessment and calibration of driver trust in automated vehicles*, T2021/5, February 2021, TRAIL Thesis Series, the Netherlands
- Schneider, F., *Spatial Activity-travel Patterns of Cyclists*, T2021/4, February 2021, TRAIL Thesis Series, the Netherlands
- Madadi, B., *Design and Optimization of Road Networks for Automated Vehicles*, T2021/3, January 2021, TRAIL Thesis Series, the Netherlands
- Krabbenborg, L.D.M., *Tradable Credits for Congestion Management: support/reject?*, T2021/2, January 2021, TRAIL Thesis Series, the Netherlands
- Castelein, B., *Accommodating Cold Logistics Chains in Seaport Clusters: The development of the reefer container market and its implications for logistics and policy*, T2021/1, January 2021, TRAIL Thesis Series, the Netherlands
- Huang, B., *The Influence of Positive Interventions on Cycling*, T2020/20, December 2020, TRAIL Thesis Series, the Netherlands
- Xiao, L., *Cooperative Adaptive Cruise Control Vehicles on Highways: Modelling and Traffic Flow Characteristics*, T2020/19, December 2020, TRAIL Thesis Series, the Netherlands
- Polinder, G.J., *New Models and Applications for Railway Timetabling*, T2020/18, December 2020, TRAIL Thesis Series, the Netherlands
- Scharpff, J.C.D., *Collective Decision Making through Self-regulation*, T2020/17, November 2020, TRAIL Thesis Series, the Netherlands
- Guo, W., *Optimization of Synchromodal Matching Platforms under Uncertainties*, T2020/16, November 2020, TRAIL Thesis Series, the Netherlands
- Narayan, J., *Design and Analysis of On-Demand Mobility Systems*, T2020/15, October 2020, TRAIL Thesis Series, the Netherlands
- Gong, X., *Using Social Media to Characterise Crowds in City Events for Crowd Management*, T2020/14, September 2020, TRAIL Thesis Series, the Netherlands