

Traffic Congestion Pattern Classification Using Multi-class SVM

Hong Nam Nguyen¹, Panchamy Krishnakumari², Hai L. Vu³ and Hans van Lint²

Abstract—Searching through and selecting data sets from large traffic databases with sensor information is often a cumbersome manual process. In this paper we present an idea that may dramatically fasten and streamline this process. The idea is to build a fast search index (COSI: CONgestion Search engIne) based on meta data in combination with features from the traffic patterns along routes. Instead of ploughing through the raw detector data, COSI makes it possible to search through higher level traffic (congestion) patterns. This paper explores a first step in developing COSI: a method to classify traffic congestion patterns reconstructed with the well known adaptive smoothing method. We demonstrate the method using a preliminary set of loop detector data in the Netherlands. We extract and store individual traffic congestion patterns of these regions as images. We then use SURF (Speeded Up Robust Features) to identify the descriptors vector of each image, and apply the bag-of-features technique to generate low-dimensional representation vectors for them. Finally, we use a multiclass extended SVM algorithm to classify these patterns. The results of the method are presented and a synthesis of the findings is given. We close with some preliminary conclusions and an outlook to the further development of COSI.

Index Terms—Traffic congestion, Support Vector Machine, feature extraction

I. INTRODUCTION

A key requirement for ITS is the availability of accessible and searchable archived data from many different types of sensors, such as induction loops, and probe vehicles. One example of an effort to make archived traffic data available to both public and private ITS service providers is the national datawarehouse of traffic information (NDW [1]); a Dutch organisation that archives and provides real-time access to traffic data from the Dutch national highway authority (Rijkswaterstaat), the 12 Dutch provinces, two metropolitan regions and four of the largest cities in the Netherlands (Amsterdam, Rotterdam, Utrecht and The Hague). From its inception in 2008 until today, the NDW historical database now comprises over 200 TB of raw traffic data (speeds, flows, travel times) collected from over 24,000 measurement sites along 7,100 kilometers of national, provincial and urban roads. The applications in which NDW data are used today range from national to local policy assessment, model development, evaluation of ITS and regional traffic management services.

A fundamental problem with the NDW database, and more generally, any large ITS data archive, is that searching

through and selecting a set of archived data is a cumbersome process. For example, these data are used to provide the input and reference data for ex ante evaluation studies with (either micro or macroscopic) traffic simulation models. In this case one may need to choose data under specific sets of circumstances or conversely, data under "normal conditions" (without incidents or other disturbances). This search process to data is essentially a manual process during which analysts have to process the raw data and inspect resulting graphs or contour plots in combination with many other data sources to judge whether a specific time on a specific location fits the search requirements.

Our objective is to largely automate this data selection process by allowing users to search through the NDW database with intuitive and "human understandable" keywords that depict either the nature of the traffic patterns they search for, or the circumstances under which these have occurred. To this end, we first propose the conceptual of a new search engine (COSI) and then focus on the development of the traffic congestion patterns classification in this paper as the first step in building such a fast index search. The proposed COSI is a new concept that there is no related transportation literature about it yet. Consider a road operator who studies the aftermath of a severe accident that temporarily blocked the entire carriageway (Fig. 1(a)). In this process, the operator wants to compare the performance of the incident management team (clearance time, resulting vehicle loss hours, etc) to similar situations in the past. Herein, we propose to develop a search engine that enables road operators to look for traffic patterns with similar characteristics (incident, temporary full carriageway blockage, vehicle loss-hours, spatio-temporal extent of the queue). Fig. 1(b) and (c) show examples of such patterns from the NDW database.

II. A CONGESTION SEARCH ENGINE

Fig. 2 sketches the envisaged layout of our traffic database facilities and the place of COSI (CONgestion Search engIne) in it. At the time of writing we already have a direct and continuous feed of raw traffic data from NDW, which contain lane / userclass specific loopdata (speeds, flows), and travel times from various types of Automatic Vehicle Identification (AVI) systems. These raw data are processed (e.g. data cleansing and missing data handling) and geo-tagged before they are stored in a large partitioned PostgreSQL database. As soon as these data are uploaded in the database, the carriageway flows and averaged speeds are computed and stored in the database for every carriageway cross section. These carriageway data form the base ingredients for COSI.

¹Ho Chi Minh University of Technology, Vietnam; Email: nhnam@hcmut.edu.vn

²Delft University of Technology, The Netherlands; Email: [\[p.k.krishnakumari,j.w.c.vanlint\]@tudelft.nl](mailto:[p.k.krishnakumari,j.w.c.vanlint]@tudelft.nl)

³Swinburne University of Technology, Australia; Email: hvu@swin.edu.au

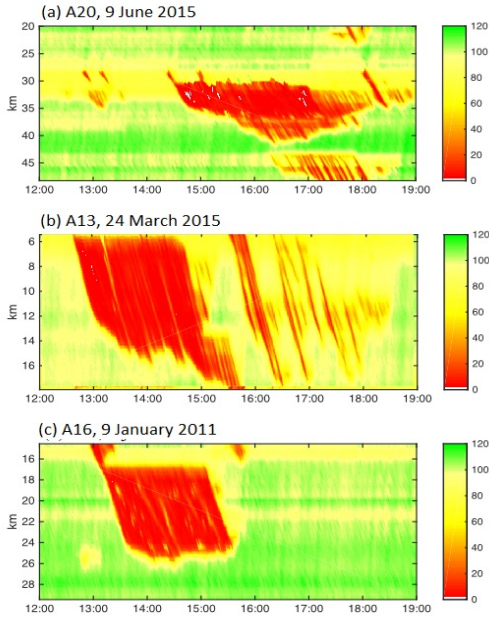


Fig. 1: Three contour maps showing severe incidents with similar characteristics in terms of e.g. incident characteristics, spatiotemporal extent of the queue, vehicle loss hours.

The overall idea of COSI is as follows, of which the bullets 1, 3 and 4 represent a substantial set of innovations that still need to be realized—we return to these in the final section.

- 1) The COSI search engine periodically (e.g. at the end of each day) searches through a carriageway level graph representation of the NDW network and identifies new congestion patterns. For this, meta data are used (traffic information, news, the 'top 100' bottlenecks, etc).
- 2) The adaptive smoothing method (ASM) [2]–[4] is used to estimate the space time patterns (e.g. Fig 1) of speed and flow along the selected routes using all available data.
- 3) The congestion patterns are classified into a set of "known" patterns, using methods of which one is developed in the remainder of this article. This process results in class labels and probabilities (or memberships in case we decide upon a fuzzy rather than probabilistic approach).
- 4) These class probabilities are augmented with meta data describing the circumstances of the congestion patterns (route graph details, date/time, weather, incidents, events, statistics such as total vehicle loss hours). The resulting combinations now form a searchable index for the underlying database

In the remainder of this paper, we will discuss a preliminary approach to the third bullet. In this approach we (a) address the problem that space time contour plots of congestion patterns will differ in (spatiotemporal and thus pixel) size; (b) develop a traffic congestion pattern classifier; and (c) discuss the mechanics and future improvements of the classification process based on our preliminary results.

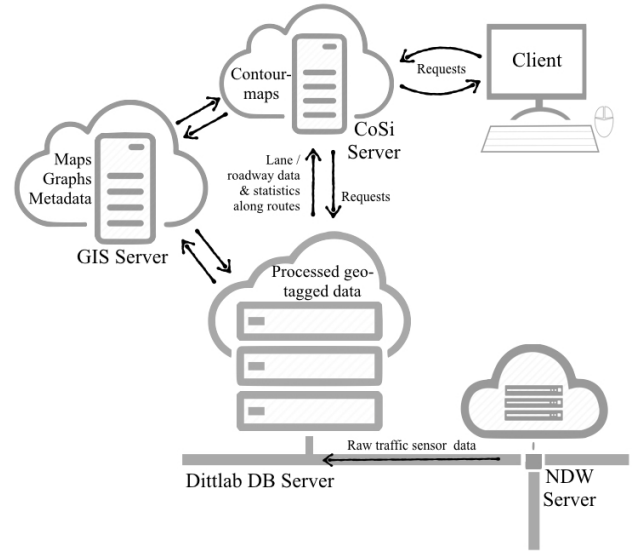


Fig. 2: Schematic overview of COSI (CONgestion Search engine)

III. METHODS

Classifying congestion patterns is not a new idea. Kerner's ASDA/FOTO method [5] is the most well-known theory-laden approach and a multitude of machine learning alternatives are available [6]–[9]. Whereas the latter focus on class labels that indicate level of severity (e.g. light, medium and heavy congestion); our aim, like Kerner's, is to classify entire spatiotemporal congestion patterns. In contrast to Kerner, we do not use an elaborated set of expert rules but flexible and modern machine learning methods. In this section we describe the methods involved in developing a first proof of concept for this advanced pattern classifier in COSI. The pattern classification process consists of three main steps: (i) detecting and extracting traffic congestion patterns (essentially images) from raw data using contour detection; (ii) building representation vectors of these traffic pattern images, (iii) classifying the set of representation vectors. Before going into details, we first briefly outline the basic concepts.

The basic ingredients for our method are contour maps of detector data generated with the adaptive smoothing method (ASM). Since this method is extensively described elsewhere, we refer the reader to e.g. [2]–[4] for details. Using the contour maps, in step (i) we extract the individual traffic congestion patterns from each contour plot—there might be multiple congestion patterns in one space-time plot. This is done using contour extraction which detects the location and duration of the congestion. As a result of this step, we have a data set of different traffic congestion pattern images. This data set is manually classified into groups depending on the size of images (space and time extent of traffic jam) and the type of congestion. In step (ii) image processing

techniques [10] are used to extract a so-called description vector of each image belonging to each predefined class. However, we cannot directly apply classification methods such as the support vector machine (SVM) [11] to this data set because it consists of description vectors of varying size. To resolve this issue, we create representation vectors from these description vectors for each image based on the bag-of-features model [12]. After this transformation step, we have a set of representation vectors which in step (iii) are used for the actual classification. Fig. 3 schematically outlines the classification process.

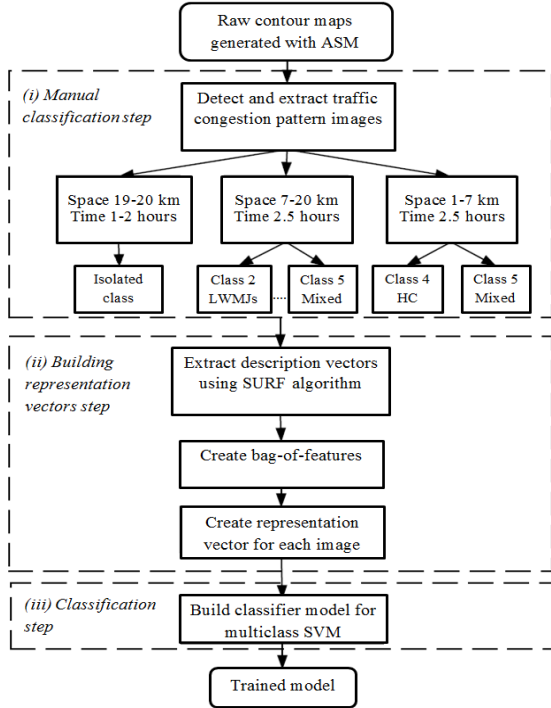


Fig. 3: Overview of our three-step approach.

A. Contour Detection

Contour detection aims at finding the edges that are connected to form a closed curve, i.e. the boundary of an object in an image. There are various methods to do this. For this paper we have implemented a relatively simple and classic algorithm. The first step is to use a Gaussian filter to average the information as we are only interested in the boundaries and not the details. Then, we use prior information about the data to filter out irrelevant information. In our case, this prior information involves using the speed limits of the highways and the assumption that low speed (e.g. < 65 km/h) implies congestion. The result is a boolean mask that sets the relevant information as 1 and irrelevant as 0. After thresholding, dilation is used to fill up the holes created so that we have a smooth mask. This smoothed mask is then used to detect the contours in the image by joining the continuous points along the boundary with similar (pixel) intensity [13]. The detected contours are used to define the

boundary to extract each pattern in an image. The result of each step is shown with a small example in Fig. 4.

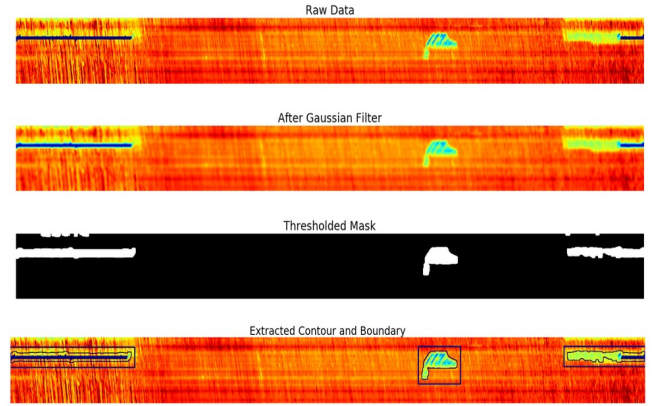


Fig. 4: Contour Extraction Process

B. Bag-of-features

The underlying idea of the bag-of-features approach is that images can be characterized by regions with large changes in contrast or luminescence, such as edges of objects. By collecting so-called key points in those regions, a compressed low-dimensional vector representing that image can be constructed. We use the so-called Laplacian of Gaussian (LoG) detector with Box filter to automatically detect key-points from images [17]. The detected key-points are depicted using the so-called SURF descriptor [10], which is a 128-dimensional real valued feature vector. The descriptors are robust against various image variations such as scale changes; affine distortions; or illumination changes. The result of the procedure is shown in Fig. 5 (top), where the key points are denoted by a circle symbol in the images. As can be expected, key-points are usually present around the edges of the traffic pattern.

Images can be represented by sets of key-point descriptors. However, the sets of key-point descriptors vary in cardinality and lack meaningful ordering. This creates difficulties for classification using support vectors machines (SVM) because this classification algorithm requires a fixed dimension of the input data set. To resolve this problem, we cluster the key-point descriptors in their feature space into a large number of clusters using a k-means clustering algorithm and encode each key-point with the index of the cluster to which it belongs. We consider each cluster as a visual feature (an "archetype" keypoint) that represents a specific local pattern shared by the key-points in the cluster. By mapping the key-points to these "archetype" features, we have a feature vector, which represents each image as a so-called "bag-of-features". From here it is easy to construct feature vectors of similar size for each images, however small or large. We do this by constructing a histogram of these k "archetype" features for every image. Put more generally, the clustering process generates a visual feature vocabulary describing different local patterns in images and the number of clusters

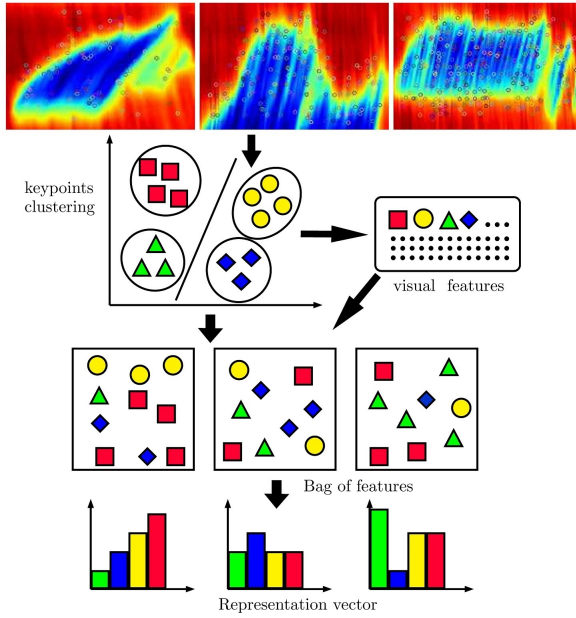


Fig. 5: Outline of feature vector (histogram) construction.

k determines the size of the resulting feature vector. The process of generating these feature vectors (histograms) is illustrated in Fig. 5.

C. Multi-class Support Vectors Machines

As we will explain further down, we will manually classify a set of "archetype" traffic patterns. To determine whether a sample pattern belongs to an archetype pattern we need a classifier. Support Vectors Machines (SVM) are among the most robust and efficient classification algorithms. The idea is to construct hyper-planes in a high-dimensional space such that example vectors $\mathbf{x}_i \in X$ (in our case the feature histograms) can be separated and labeled to a corresponding class label $y_i \in \mathbf{y}$. The supervised learning method that finds these hyper-planes for a given data set (X, \mathbf{y}) such that the minimum distance that separates the hyper-plane to the nearest example is maximized. The basic SVM [11] is designed for binary classification. How to effectively extend it for multi-class classification is still an ongoing research problem, however, there is a simple and highly effective method which appears to be very difficult to beat in terms of accuracy and computational speed. This approach is the *one-versus-all* (OvA) method [14], [15]. The idea is intuitive and easy to implement. In the OvA method, an individual SVM model is created for each of the multiple output classes. Each of these classifiers identifies an individual class against its complement (all other classes in the model) as though it were a binary classification issue. Let us assume that the training algorithm for binary classification is learner \mathbf{L} . The training algorithm for an OvA learner constructed from learner \mathbf{L} is as follows [15]:

Input: \mathbf{L} is learner (training algorithm for binary classifiers); samples X ; labels \mathbf{y} where $y_i \in \{1, \dots, K\}$ is the class label of \mathbf{x}_i ;

Output: a list of classifiers model f_k for $k \in \{1, \dots, K\}$

LOOP Process

for $k = 1$ to K **do**

Construct a new label vector \mathbf{y} :

$$y_i = \begin{cases} 1 & i = k \\ 0 & i \neq k \end{cases} \quad (1)$$

Classifier model $f_k \leftarrow$ apply learner \mathbf{L} to X , vector \mathbf{y}

end for

Making decisions means applying the classifier to an unknown sample \mathbf{x}_j and predicting the label \hat{y}_j for which the corresponding classifier reports the highest confidence score:

$$\hat{y}_j = \arg \max_{k \in 1 \dots K} f_k(\mathbf{x}_j) \quad (2)$$

Note that also other decision rules than depicted in equation (2) can be used, we return to this in the last section.

IV. EXPERIMENTAL SETUP

A. Data

The data used for the experiments were collected from two heavily congested roads in the Netherlands:

- The A13 Southbound from Den Haag-Zuid to Rotterdam center.
- The A15 Eastbound from Havens 5500-5700 to Rotterdam IJsselmonde.

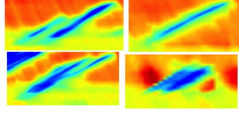
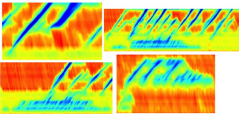
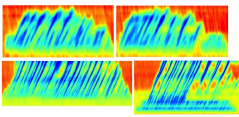
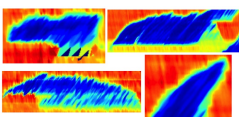
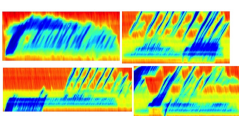
TABLE I: Number of traffic patterns in each class

Class	Space (S) km	Time (T) minutes	Number of patterns
Isolated	$S \approx 20$	$75 < T < 150$	32
Low frequency	$7 < S < 20$	$T \approx 160$	34
High Frequency	$7 < S < 13$	$T \approx 160$	18
Homogeneous	$1.3 < S < 20$	$T \approx 160$	19
Mixed	$6 < S < 20$	$T \approx 160$	18

Space-time plots of carriageway speeds for these two roads were constructed using all available loop data for the entire month of March 2015. The loop detectors are on average 500 meters apart and collect 1 minute average speeds. Each space-time plot represents a period of 23hours and 58minutes from 00:01 to 23:59 at a resolution of 30s and 100m—the ASM essentially allows us to create a contour plot at any desired resolution, for details we again refer to [2]–[4]). Each of the plots contains multiple congestion patterns. After identifying and extracting each pattern separately, there were 140 traffic congestion patterns detected on the first road and 160 patterns on the second road. As we were interested only in large scale patterns (which we arbitrarily consider to span over 1 km and 75 minutes—see Table I) with similar space-time ratios, 121 patterns were selected to create a first training set for the classification process. We manually classified the patterns into five different classes based on their spatio-temporal extent and the characteristics of traffic congestion. Table II shows the five classes with some examples we identified. This classification of course is arbitrary—other analysts may have come up with more or less classes

and different criteria. Table I contains information about the space, time extent of the congestion and number of patterns in each class. Note that the number of patterns per class is small; this first trial is intended to demonstrate the ideas and learn lessons for larger-scale application.

TABLE II: Different classes of traffic patterns

Name	Remarks	Examples
Isolated Wide Moving Jams (WMJ)	Typically resulting from large disturbances (e.g. abrupt braking) further downstream	
Heterogeneous Congestion I - Low Frequency WMJ's	Large-scale light congestion patterns with a few WMJ's emitting from the congested area	
Heterogeneous Congestion II - High Frequency WMJ's	Large-scale light congestion patterns with many WMJ's emitting from the congested area	
Homogeneous Congestion	High-density severe congestion, typically due to incidents or other lane blockages	
Mixed Large Scale Pattern	Combination patterns not falling in either of the other categories	

B. Performance Indicators

The accuracy of the SVM OVA classifier was measured using the following formula:

$$accuracy = \frac{\text{Number of correctly classified images}}{\text{Total number of images}} \quad (3)$$

The overall accuracy of the algorithm was also measured using the above equation through leave-one-out cross validation. The whole data set was divided into one test data and the rest as the training data and the accuracy of the test data classification was measured. This was repeated for each image in the training data set to build a confusion matrix. A confusion matrix (also called a contingency table) [16] is constructed to help us investigate which class is behaving poorly and study that class further to understand the data better in order to make future recommendations for improved accuracy.

V. RESULTS

This section presents the results of the proposed method. The leave-one-out cross validation achieved an average prediction accuracy of 94%. A more elaborate confusion matrix

is given in the Table III. The table shows the percentage of the patterns in each class that have been classified correctly and erroneously given the ground truth. For example, for the Isolated Wide Moving Jams class, all the ground truth patterns in that class were correctly classified with 100% accuracy, but there were some patterns in other classes which were wrongly classified as Isolated. The Homogeneous Congestion class contains the most patterns that were wrongly classified with 78% accuracy. Some of the patterns in this class was classified as Isolated class, hence the less accuracy. In the next section, we will discuss these wrongly classified patterns and the reason behind it.

TABLE III: Leave-one-out cross validation results

KNOWN	PREDICTED				
	Isolated	Low Frequency	High Frequency	Homogeneous	Mixed
Isolated	1.00	0.00	0.00	0.00	0.00
Low Frequency	0.03	0.94	0.00	0.00	0.03
High Frequency	0.00	0.00	1.00	0.00	0.00
Homogeneous	0.22	0.00	0.00	0.78	0.00
Mixed	0.00	0.00	0.00	0.00	1.00

VI. SYNTHESIS

Let us first briefly compare our results to the literature. For example, in [6] an accuracy of 95.28% was achieved for classifying highway traffic congestion using motion vector statistical properties; in [7] a score of 94.50% was achieved, where they used so-called "holistic properties" to classify traffic congestion; and in [8] the reported score was 91.29%. First of all, these papers used substantially larger data sets. More importantly, these classification exercises were of a very different nature than the one presented in this paper. They classify/rate the overall traffic congestion levels (light, medium, heavy). Our objective is to classify entire spatiotemporal patterns that may encompass any number of complex traffic states.

From the Table III, we can see that two classes have low accuracy compared to the other classes, namely Low Frequency WMJ and Homogeneous congestion. We hypothesize this is due to the small training data set and/or the limited representativeness of the samples in that class. We then investigate deeper into the data to qualitatively test this hypothesis.

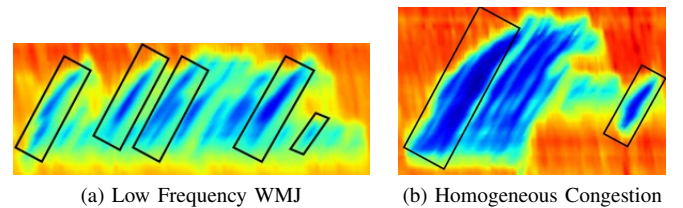


Fig. 6: Example of wrongly classified patterns (1): (a) patterns in the LF-WMJ class; (b) pattern in the HC class. Both (a) and (b) were classified in the Isolated WMJ class

Two of the wrongly classified patterns are shown in Fig. 6; in this case both are wrongly classified as Isolated WMJ instead of Low Frequency WMJ and Homogeneous Congestion respectively. It can be observed that these patterns are in fact constituted of Isolated WMJ patterns (compare the patterns highlighted with the rectangle in Fig. 6 with Table II). One complex pattern may contain different types of other classes. Clearly, successful classification depends on the (highly) subjective manual classification process, and the degree in which we succeed in labeling patterns that can be distinguished through OvA SVM. This is confirmed by a second set of wrongly classified patterns in Fig. 7. We can clearly see why the classifier would confuse these Homogeneous Congestion patterns as Isolated WMJ. It is because they indeed have similar characteristics as the isolated pattern.

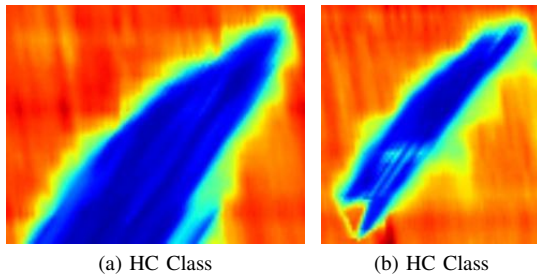


Fig. 7: Example of wrongly classified patterns (2): Both (a) and (b) are patterns in the HC class. However, they are wrongly classified as patterns in the Isolated WMJ class

VII. CONCLUSION AND FURTHER RESEARCH

In this paper, we proposed a methodology, consisting of contour detection, the bag-of-features method using SURF descriptors and a multiclass OvA SVM classifier, to automatically classify spatiotemporal traffic patterns using network sensor data. The different components proved sufficiently adequate to label complex traffic patterns with an acceptable accuracy, although the data set we used was too small to warrant definite conclusions. By studying the few wrongly classified patterns it became clear that much of the ambiguity is due to the subjective manual labeling to construct the training data.

This ambiguity, however, is unavoidable and does not pose an obstacle to develop the overall idea of a congestion search engine (COSI). In fact, there are many future directions of research that directly relate to it. First, we can make use of the ensemble scores to directly put a confidence score to each classification. Further sophistication can be reached by ensemble bootstrapping or more modern Bayesian techniques. Second, we envisage an iterative manual-automated classification procedure in which we re-evaluate the manual classification after each training round using the classification scores of the OvA SVM. This may yield splitting or combining classes; hierarchically subdividing classes or otherwise. Third, we will combine meta data (type of date/time, circumstances, topological characteristics, etc) to assist in

classifying the patterns better. Another extension will be to combine speed plots with flow contour plots for better definition of different classes of congestion and for improved accuracy. We will also explore the use of shape features instead of pixel intensity features to define the description vectors for the classification. Finally, we will continue to enrich our database with more congestion patterns and aim towards identifying more complex patterns.

ACKNOWLEDGMENT

The research in this paper is sponsored by the Dutch National Datawarehouse of Traffic Information [1]. We thank the anonymous reviewers for their many constructive remarks that enabled us to improve the paper significantly.

REFERENCES

- [1] National Datawarehouse of Traffic Information, <http://www.ndw.nu/en/>
- [2] T. Schreiter, H. Van Lint, M. Treiber, and S. Hoogendoorn, "Two fast implementations of the adaptive smoothing method used in highway traffic state estimation," in *13th International IEEE Conference on Intelligent Transportation Systems*, Funchal; Portugal, pp. 1202-1208, 2010.
- [3] M. Treiber and D. Helbing, "Reconstructing the Spatio-Temporal Traffic Dynamics from Stationary Detector Data," *Cooperative Transportation Dynamics*, vol. 1, pp. 3.1-3.24, 2002.
- [4] van Lint, J. W. C. and S. P. Hoogendoorn, "A Robust and Efficient Method for Fusing Heterogeneous Data from Traffic Sensors on Free-ways," *Computer-Aided Civil and Infrastructure Engineering*, 25(8): 596-612, 2010.
- [5] B. S. Kerner, H. Rehborn, M. Aleksic, and A. Haug, "Recognition and tracking of spatial-temporal congested traffic patterns on freeways," *Transportation Research Part C: Emerging Technologies*, vol. 12, pp. 369-400, 2004.
- [6] A. Riaz, S.A. Khan, "Traffic congestion classification using motion vector statistical features", *Proc. SPIE 9067, Sixth International Conference on Machine Vision (ICMV 2013)*, 2013.
- [7] A. Sobral, L. Oliveira, L. Schnitman, "Highway traffic congestion classification using holistic properties", *10th IASTED International Conference on Signal Processing, Pattern Recognition and Applications*, 2013.
- [8] T. Thianniwet, S. Phosaard, W. Pattara-Atikom, "Classification of Road Traffic Congestion Levels from Vehicle's Moving Patterns: A Comparison Between Artificial Neural Network and Decision Tree Algorithm", *Electronic Engineering and Computing Technology*, Vol. 60, pp. 261-271, 2010.
- [9] X. Jin, D. Srinivasan, and R. L. Cheu, "Classification of freeway traffic patterns for incident detection using constructive probabilistic neural networks," *IEEE Transactions on Neural Networks*, vol. 12, pp. 1173-1187, 2001.
- [10] Bay, H., Tuytelaars, T., and Van Gool, L., "SURF: Speeded Up Robust Features", *In Proceedings of the Ninth European Conference on Computer Vision*, 2006
- [11] C. Cortes and V. Vapnik. Support-vector network. *Machine Learning*, pp.273-297, 1995.
- [12] S. O'Hara and B.A. Draper, "Introduction to the bag of features paradigm for image classification and retrieval" *Arxiv preprint arXiv:1101.3354*, 2011
- [13] S. Suzuki, "Topological structural analysis of digitized binary images by border following", *Computer Vision, Graphics, and Image Processing* 30.1, pp. 32-46, 1985.
- [14] Vapnik, V. N. and Vapnik, V., "Statistical learning theory", Vol. 1. New York: Wiley, 1998.
- [15] Crammer, K. and Singer, Y, "On the algorithmic implementation of multiclass kernel-based vector machines.", *The Journal of Machine Learning Research* 2, pp. 265-292, 2002.
- [16] Fawcett T. "An introduction to ROC analysis". *Pattern recognition letters*, 27(8):861-74, 2006.
- [17] Canny, John. "A computational approach to edge detection." *IEEE Transactions on pattern analysis and machine intelligence*, pp. 679-698, 1986.