Automating Near-Miss Crash Detection Using Existing Traffic Cameras

Final Report
March 2019

ctre
Center for Transportation
Research and Education

IOWA STATE UNIVERSITY
Institute for Transportation

Sponsored by
Iowa Department of Transportation
(InTrans Project 17-619)

About InTrans and CTRE

The mission of the Institute for Transportation (InTrans) and Center for Transportation Research and Education (CTRE) at Iowa State University is to develop and implement innovative methods, materials, and technologies for improving transportation efficiency, safety, reliability, and sustainability while improving the learning environment of students, faculty, and staff in transportation-related fields.

Disclaimer Notice

The contents of this report reflect the views of the authors, who are responsible for the facts and the accuracy of the information presented herein. The opinions, findings and conclusions expressed in this publication are those of the authors and not necessarily those of the sponsors.

The sponsors assume no liability for the contents or use of the information contained in this document. This report does not constitute a standard, specification, or regulation.

The sponsors do not endorse products or manufacturers. Trademarks or manufacturers' names appear in this report only because they are considered essential to the objective of the document.

ISU Non-Discrimination Statement

Iowa State University does not discriminate on the basis of race, color, age, ethnicity, religion, national origin, pregnancy, sexual orientation, gender identity, genetic information, sex, marital status, disability, or status as a U.S. veteran. Inquiries regarding non-discrimination policies may be directed to Office of Equal Opportunity, 3410 Beardshear Hall, 515 Morrill Road, Ames, Iowa 50011, Tel. 515 294-7612, Hotline: 515-294-1222, email eooffice@iastate.edu.

Iowa DOT Statements

Federal and state laws prohibit employment and/or public accommodation discrimination on the basis of age, color, creed, disability, gender identity, national origin, pregnancy, race, religion, sex, sexual orientation or veteran's status. If you believe you have been discriminated against, please contact the Iowa Civil Rights Commission at 800-457-4416 or the Iowa Department of Transportation affirmative action officer. If you need accommodations because of a disability to access the Iowa Department of Transportation's services, contact the agency's affirmative action officer at 800-262-0003.

The preparation of this report was financed in part through funds provided by the Iowa Department of Transportation through its "Second Revised Agreement for the Management of Research Conducted by Iowa State University for the Iowa Department of Transportation" and its amendments.

The opinions, findings, and conclusions expressed in this publication are those of the authors and not necessarily those of the Iowa Department of Transportation.

Technical Report Documentation Page

1. Report No.	2. Government Accession No.	3. Recipient's Catalog No.
InTrans Project 17-619		

4. Title and Subtitle	5. Report Date
Automating Near-Miss Crash Detection Using Existing Traffic Cameras	March 2019
	6. Performing Organization Code

7. Author(s)	8. Performing Organization Report No.
Anuj Sharma (orcid.org/0000-0001-5929-5120), Pranamesh Chakraborty (orcid.org/0000-0003-2624-5543), Neal Hawkins (orcid.org/0000-0003-0618-6275), and Skylar Knickerbocker (orcid.org/0000-0002-0202-5872)	InTrans Project 17-619

9. Performing Organization Name and Address	10. Work Unit No. (TRAIS)
Center for Transportation Research and Education Iowa State University 2711 South Loop Drive, Suite 4700 Ames, IA 50010-8664	11. Contract or Grant No.

12. Sponsoring Organization Name and Address	13. Type of Report and Period Covered
Iowa Department of Transportation 800 Lincoln Way Ames, IA 50010	Final Report
	14. Sponsoring Agency Code TSIP

15. Supplementary Notes

Visit www.intrans.iastate.edu for color pdfs of this and other research reports.

16. Abstract

State departments of transportation (DOTs) typically install a relatively large number of cameras across freeways for surveillance purposes. It is estimated that there will be approximately a billion of these traffic cameras worldwide by 2020. However, most of these cameras are used for manual surveillance purposes only. Hence, there is a significant need to develop automatic anomaly detection algorithms that use the data from these cameras.

This study was divided into two broad topics involving the detection of freeway traffic anomalies from cameras: detecting traffic congestion from camera images and detecting traffic incidents from camera videos. Two modern deep learning techniques, the traditional deep convolutional neural network (DCNN) and the you only look once (YOLO) models, were used to detect traffic congestion from camera images and compared with a shallow model, support vector machine (SVM).

The YOLO model achieved the highest accuracy of 91.2%, followed by the DCNN model with an accuracy of 90.2%; 85% of images were correctly classified by the SVM model. The deep models were found to perform well in challenging conditions too, such as nighttime or poor weather conditions.

To detect traffic incidents from camera videos, the research team proposed a semi-supervised approach for detecting incident trajectories. Results showed that the proposed semi-supervised trajectory classification outperformed the traditional semi-supervised techniques and its supervised counterpart by a significant margin.

17. Key Words	18. Distribution Statement
computer vision—deep learning-anomaly detection—traffic cameras—traffic congestion	No restrictions.

19. Security Classification (of this report)	20. Security Classification (of this page)	21. No. of Pages	22. Price
Unclassified.	Unclassified.	53	NA

Form DOT F 1700.7 (8-72) **Reproduction of completed page authorized**

AUTOMATING NEAR-MISS CRASH DETECTION USING EXISTING TRAFFIC CAMERAS

Final Report
March 2019

Principal Investigator
Anuj Sharma, Research Scientist
Center for Transportation Research and Education, Iowa State University

Co-Principal Investigators
Neal Hawkins, Associate Director
Institute for Transportation, Iowa State University

Skylar Knickerbocker, Research Engineer
Center for Transportation Research and Education, Iowa State University

Research Assistant
Pranamesh Chakraborty

Authors
Anuj Sharma, Pranamesh Chakraborty, Neal Hawkins, and Skylar Knickerbocker

Sponsored by
Iowa Department of Transportation
(InTrans Project 17-619)

Preparation of this report was financed in part
through funds provided by the Iowa Department of Transportation
through its Research Management Agreement with the
Institute for Transportation

A report from
Institute for Transportation
Iowa State University
2711 South Loop Drive, Suite 4700
Ames, IA 50010-8664
Phone: 515-294-8103 / Fax: 515-294-0467
www.intrans.iastate.edu

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGMENTS

The authors would like to thank the Iowa Department of Transportation (DOT), which used Traffic Safety Improvement Program (TSIP) funding for this project.

Previous versions of Chapters 3 and 4 of this report were published in the following journal and conference proceedings, respectively:

Chakraborty, P., Y. O. Adu-Gyamfi, S. Poddar, V. Ahsani, A. Sharma, and S. Sarkar. 2018a. Traffic Congestion Detection from Camera Images Using Deep Convolution Neural Networks. *Transportation Research Record: Journal of the Transportation Research Board*, Vol. 2672, No. 45, pp. 222–231. doi:10.1177/0361198118777631.

Chakraborty, P., A. Sharma, and C. Hegde. 2018b. Freeway Traffic Incident Detection from Cameras: A Semi-Supervised Learning Approach. In *IEEE Conference on Intelligent Transportation Systems, Proceedings, ITSC*. doi:10.1109/ITSC.2018.8569426.

EXECUTIVE SUMMARY

The number of internet-connected cameras is increasing at a rapid pace. State departments of transportation (DOTs) typically install a large number of closed-circuit television (CCTV) cameras across freeways for surveillance purposes. However, it is virtually impossible to manually monitor such a large network of cameras constantly. Hence, there is a significant need to develop automatic anomaly detection algorithms that use the data from these cameras.

This study was divided into two broad topics involving the detection of freeway traffic anomalies from cameras: detecting traffic congestion from camera images and detecting traffic incidents from camera videos.

The first research objective involved detecting traffic congestion from camera images. Two modern deep learning techniques, the traditional deep convolutional neural network (DCNN) and you only look once (YOLO) models, were used to detect traffic congestion from camera images. A shallow model, support vector machine (SVM) was also used for comparison and to determine the improvements that might be obtained using costly GPU techniques

The YOLO model achieved the highest accuracy of 91.2%, followed by the DCNN model with an accuracy of 90.2%; 85% of images were correctly classified by the SVM model. Congestion regions located far away from the camera, single-lane blockages, and glare issues were found to affect the accuracy of the models. Sensitivity analysis showed that all of the algorithms were found to perform well in daytime conditions, but nighttime conditions were found to affect the accuracy of the vision system. However, for all conditions, the areas under the curve (AUCs) were found to be greater than 0.9 for the deep models. This result shows that the models performed well in challenging conditions as well.

The second part of this study aimed at detecting traffic incidents from CCTV videos. Typically, incident detection from cameras has been approached using either supervised or unsupervised algorithms. A major hindrance in the application of supervised techniques for incident detection is the lack of a sufficient number of incident videos and the labor-intensive, costly annotation tasks involved in the preparation of a labeled dataset.

In this study, the research team approached the incident detection problem using semi-supervised techniques. Maximum likelihood estimation-based contrastive pessimistic likelihood estimation (CPLE) was used for trajectory classification and identification of incident trajectories. Vehicle detection was performed using state-of-the-art deep learning-based YOLOv3, and simple online real-time tracking (SORT) was used for tracking. Results showed that CPLE-based trajectory classification outperformed the traditional semi-supervised techniques (self learning and label spreading) and its supervised counterpart by a significant margin.

CHAPTER 1. INTRODUCTION

Traffic congestion on freeways poses a major threat to the economic prosperity of the nation (Owens et al. 2010). Freeway congestion is usually classified into two categories: recurrent congestion and non-recurrent congestion (Ozbay and Kachroo 1999, Dowling et al. 2004, Anbaroglu et al. 2014). Recurrent congestion typically exhibits a daily pattern, observed in morning or evening peaks, while non-recurrent congestion is mainly caused by unexpected events or anomalies such as traffic incidents or stalled vehicles (Anbarogluet al. 2014). Such events are a major source of travel time variability (Noland and Polak 2002) and often cause frustration to commuters (TTI with Cambridge Systematics 2005). Hence, traffic anomaly detection has been identified as a crucial method for the reduction of non-recurrent traffic congestion (Sussman 2005). Early incident detection was shown to save 143.3 million man-hours and $3.06 million in 2007 (Schrank and Lomax 2007). Consequently, significant research has been done on the development of accurate anomaly detection algorithms.

With the widespread use of mobile phones and video surveillance systems, incident detection time has been reported to have decreased significantly in recent years, particularly in urban conditions. In general, it has been found that incidents are usually reported within 2 minutes and seldom within more than 5 minutes (Yang et al. 2018). However, this reporting mostly relies on either calls from people directly involved in the incidents or manual inspection of hundreds of cameras installed on the freeways, which hinders the scalability and reliability of the detection system. Cameras, however, when used for automatically detecting traffic anomalies, can report such anomalies within seconds. In the 2018 AI City Challenge, traffic incident detection times were reported to be within 3 to 10 seconds (Naphade et al. 2018).

With the recent advancements in deep learning techniques and improvements in object detection accuracies from videos and images (Han et al. 2018), cameras installed on freeways can be used to automatically detect traffic anomalies in significantly less time than other data sources. State departments of transportation (DOTs) typically install these roadside cameras on freeways and arterials for surveillance tasks such as incident detection. These cameras, when used effectively, can serve as useful sources for detecting traffic anomalies.

However, two major challenges arise in using cameras for traffic anomaly detection. First, these cameras are used by traffic incident managers, who can zoom, tilt, and pan the cameras according to their need. Such frequent camera movement can alter the default calibrations and thereby impact the performance of anomaly detection algorithms. Therefore, anomaly detection algorithms should not rely on the camera calibration and should be able to account for frequent camera movements. In this study, the research team developed anomaly detection algorithms that do not require camera calibration and, hence, can account for frequent movements of the cameras. Secondly, cameras are known to perform poorly in difficult weather conditions, such as snow or rain. To mitigate such disadvantages, state-of-the-art deep learning-based object detection and tracking algorithms were used in this study that can perform fairly well in adverse weather conditions.

This study is divided into two broad research objectives: detecting traffic congestion from camera images and detecting traffic incidents from camera videos using semi-supervised learning. Since traffic congestion is a major source of travel time variability, the first focus of this study was to detect congested regions, irrespective of whether the congestion observed is recurrent or non-recurrent. Camera images captured from different locations and orientations and in different weather conditions were used to successfully detect traffic congestion. Three different models were used for congestion detection tasks. Two of these were deep neural network models: deep convolutional neural networks (DCNNs) and you only look once (YOLO) (Redmon and Farhadi 2017). Because these models require time-consuming and costly graphics processing unit (GPU) training, support vector machine (SVM), a shallow learning model, was used as a comparison to determine the advantages of using the deep models.

The second part of this study went one step beyond congestion detection from images to detect traffic incidents from videos. The research team adopted a semi-supervised learning approach for trajectory classification to detect traffic incidents from videos. Traffic incident detection from videos using trajectory information comprises three basic tasks: vehicle detection, vehicle tracking and trajectory formation, and trajectory classification. In this study, a deep learning-based object detector, YOLO (Redmon and Farhadi 2018), was used to detect vehicles in video frames and a Kalman filtering-based tracker (Bewley et al., n.d.) was used to track the vehicles. Finally, the trajectories were classified into incidents and non-incidents using a semi-supervised classifier. This helped to do away with the step of manually annotating vehicle tracks in a video stream to prepare a training dataset, which is extremely labor-intensive, expensive, and not scalable. The experimental results using traffic video data provided by the Iowa DOT demonstrate that the developed framework achieves superior performance compared to supervised learning techniques with a comparable number of labeled examples.

This report is organized as follows. Chapter 2 provides a brief description of the relevant literature on traffic congestion and anomaly detection on freeways. Chapter 3 provides a detailed description of the first objective of this study, the detection of traffic congestion from camera images. The details of the methodology, data, and results and a short conclusion for the problem are provided in this chapter. Similarly, Chapter 4 describes the details of the methodology, data, and results and a brief conclusion for the second research objective, semi-supervised-based trajectory classification for incident detection from traffic camera videos. Finally, Chapter 5 provides a conclusion on the work done in this study, the limitations of the study, and the scope of future work.

CHAPTER 2. LITERATURE REVIEW

Significant research has been performed on using traffic cameras for detecting traffic anomalies and estimating traffic states, which can be used to detect traffic congestion. This section gives a brief overview of the research performed in each of these areas.

2.1. Traffic Congestion Detection from Closed-Circuit Television (CCTV) Images

Dissemination of real-time traffic information to road users can significantly improve the efficiency of traffic networks. Hence, estimating real-time traffic states and thereby detecting network anomalies, such as congestion and incidents, have been of significant interest to researchers for the last few decades.

Traditionally, traffic state estimation is conducted using point-based sensors, including inductive loops, piezoelectric sensors, and magnetic loops (Kotzenmacher et al. 2004). Recent advances in active infrared/laser radar sensors have led to these devices gradually replacing the traditional point-based sensors (Zhong and Liu 2007). Also, with the increasing use of navigation-based global positioning system (GPS) devices, probe-based data are emerging as a cost-effective way to collect network-wide traffic data (Feng et al. 2014). Video monitoring and surveillance systems also are used for gathering real-time traffic data (Ozkurt and Camci 2009). Recent advances in image processing techniques have improved the accuracy of vision-based detection. Deep learning methods, such as convolutional neural networks (CNNs), have been able to achieve human-level accuracy in image classification tasks (He et al. 2016). The basic advantage of these methods is that they do not require the identification of hand-crafted features and hence can do away with the painstaking calibration tasks needed when using camera images for traffic state estimation (Bauzaet al. 2010).

Studies have also been performed fusing multiple sources of data for traffic state estimation. Van Lint and Hoogendoorn (2010) used an extended generalized Treiber-Helbing filter for fusing probe-based and sensor-based data. Choi and Chung (2010) used fuzzy regression and Bayesian pooling techniques for estimating link travel times from probe data and sensor data. Bachmann et al. (2013) investigated several multi-sensor data fusion-based techniques to compare their ability to estimate freeway traffic speed. State DOTs have also traditionally used sensor data and probe vehicle data for traffic state estimation. However, they have also installed a large number of roadside cameras on freeways and arterials for surveillance tasks such as incident detection. These cameras are used by traffic incident managers, who can zoom, tilt, and pan the cameras according to their need. Hence, the use of cameras for traffic state estimation or congestion detection introduces challenges related to frequent camera movement, which can alter the cameras' default calibrations. However, algorithms should not rely on the exact placement of cameras and should be able to accurately detect traffic conditions for different placement scenarios.

In the present study, camera images taken from different locations and orientations and in different weather conditions were used to successfully detect traffic congestion. Three different models were used for congestion detection tasks. Two of these were deep neural network

models: DCNNs and YOLO. Because these models require time-consuming and costly GPU training, SVM, a shallow learning model, was used as a comparison to determine the advantages of using the deep models.

During the last few decades, significant research efforts have been devoted to using closed-circuit television (CCTV) cameras to determine real-time traffic parameters such as volume, density, and speed (Darwish and Abu Bakar 2015). The methods in these studies can be broadly divided into three categories: detection-based methods, motion-based methods, and holistic approaches.

Detection-based methods use individual video frames to identify and localize vehicles and thereby perform a counting task. Ozkurt and Camci (2009) used neural network methods to perform vehicle counting and classification tasks from video records. Kalman filter-based background estimation has also been used to estimate vehicle density (Balcilar and Sönmez 2008). However, these methods were found to perform poorly for videos with low resolution and high occlusion. Recent achievements in deep learning methods for image recognition tasks have led to several such methods being used for traffic counting tasks. Adu-Gyamfi et al. (2017) used DCNNs for vehicle category classification. Oñoro-Rubio and López-Sastre (2016) used two variations of CNN, namely counting CNN and hydra CNN, to conduct vehicle counting and predict traffic density. Recently, Zhang et al. (2015) used both deep learning and optimization-based methods to perform vehicle counts from low-frame-rate, high-occlusion videos.

Several motion-based methods have been proposed in the literature to estimate traffic flow using vehicle tracking information. Asmaa et al. (2013) used microscopic parameters extracted using motion detection in a video sequence. However, these methods tend to fail due to the lack of motion information and the low frame rates of videos; some vehicles appear only once in a video, and hence it is difficult to estimate their trajectories.

Holistic approaches avoid the segmentation of each object. Rather, an analysis is performed on the whole image to estimate the overall traffic state. Gonçalves et al. (2012) classified traffic videos into different congestion types using spatiotemporal Gabor filters. Lempitsky and Zisserman (2010) performed a linear transformation on each pixel feature to estimate the object density in an image; however, this approach was found to perform poorly in videos with a broad perspective. Further, both of these methods require manual annotation of each object in the images to perform the training for the counting task.

Overall, significant studies have been conducted in the past using various deep and shallow learning models to implement vehicle counting tasks and thereby determine congestion states. In the present study, the research team adopted the holistic approach to label an image as either congested or non-congested. Instead of counting each vehicle to determine the congestion state, each image was assigned its label based on nearby benchmark sensors, and then the classification task was performed. The following section provides a detailed description of the studies relevant to the second research objective of this study, the use of a semi-supervised learning approach for freeway incident detection from videos.

2.2. Freeway Incident Detection from CCTV Videos

Approaches to traffic incident detection from CCTV cameras can be broadly classified into two categories: explicit event recognition and anomaly detection.

In explicit event recognition, the explicit knowledge of the events to be identified is used for incident detection. This technique requires a priori knowledge of all of the recognizable events, which the associated automatic incident detection (AID) systems use as predefined templates to parse the incoming data for incident detection. For example, Ravinder et al. (2008) applied video image processing for traffic management to detect cases of non-adherence to lane discipline. Sadeky et al. (2010) used logistic regression over histogram of flow gradients (HFG) to determine the probability of occurrence of an accident in a video sequence. Hui et al. (2014) used a Gaussian mixture model (GMM) to detect vehicles in traffic and track them using a mean shift algorithm. Traffic incident alarms were triggered when the velocity or acceleration of the detected vehicles exceed a pre-determined threshold. Ren et al. (2016) used video-based detection to analyze the traffic state distribution characteristics in a cluster of cells dividing the lanes of a road segment.

The other popular approach to incident detection is anomaly detection. In this approach, the system attempts to learn "typical" patterns in the incoming data; any irregularities in the observed data can be classified as an incident. For example, Lou et al. (2002) used dynamic clustering techniques to cluster normal trajectories and detect abnormal ones. Piciarelli et al. (2008) used one-class SVM to detect anomalous trajectories. More recently, Yuan et al. (2017) performed anomaly detection in traffic scenarios using spatially aware motion reconstruction. Such unsupervised modeling of video sequences has also traditionally involved a combination of sparse coding and bag-of-words (BoW) models (Zhao et al. 2011). However, recent developments in deep learning techniques have resulted in new methods of learning normal video patterns and thereby detecting anomalies based on reconstruction error.

Hasan et al. (2016) used a fully convolutional feed-forward autoencoder to learn the spatio-temporal local features in videos and thereby learn the temporal regularity in video sequences. Chong and Tay (2017) also used a combination of spatial feature extractor and temporal sequencer techniques based on a convolutional long short-term memory (ConvLSTM) network for anomaly detection in videos.

The above two categories of traffic incident detection approaches can broadly be termed as supervised and unsupervised learning techniques. While supervised techniques can, in general, provide better results in detection or classification tasks, the main hindrance in their application is the scarcity of supervised data samples and the cost of manually annotating and labeling the dataset. In particular, manually annotating vehicle tracks in a video stream is extremely labor-intensive, expensive, and not scalable. In the present study, the research team established a new learning framework for traffic incident detection using recent advances in semi-supervised learning (Loog 2016). This framework can achieve the "best of both worlds." A small sample of normal vehicle tracks and the tracks of vehicles involved in an incident were manually annotated, and all other (unlabeled) vehicle tracks were used to improve the performance of the

classification. Since this approach uses trajectory-based classification for detecting incident trajectories, the first two steps are to detect vehicles and track them to assign the trajectories. A brief overview of past research on vehicle detection and tracking is provided in the following section.

Object Detection

In recent years, the evolution of CNN has resulted in significant improvements in the performance of object detection and classification. Results of the ImageNet Large Scale Visual Recognition Challenge (ILSVRC) point to dramatic improvements in object detection, localization, and classification (Russakovsky et al. 2015).

Region-based convolutional neural networks (R-CNNs) were among the first modern developments in CNN-based detection (Girshick et al. 2014). These developments involved cropping externally computed box proposals from an input image and running a neural net classifier on these crops. However, overlapping crops led to significant duplicate computations, which, in turn, led to low processing speeds. The development of Fast R-CNN involved pushing the entire input image only once through a feature extractor and cropping from an intermediate layer (Girshick 2015). This led to the crops sharing the computation load for feature extraction and thereby increased processing speed.

Recent work has focused on generating box proposals using neural networks instead of relying on the external box proposals used in R-CNN and Fast R-CNN (Szegedy et al. 2013, Erhan et al. 2014, Ren et al. 2017, Redmon et al. 2016). Such approaches involve overlaying a collection of boxes on the image at different locations, aspect ratios, and scales. These boxes are called anchors or priors. Training is then performed to predict the discrete class of each anchor and the offset by which the anchor needs to be shifted to fit the ground truth bounding box. The accuracy and computation time of the object detection algorithm depends significantly on the choice of these anchors.

The following sections discuss four recent architectures for object detection and classification: Faster R-CNN (Ren et al. 2017), single-shot multibox detector (SSD) (Liu et al. 2016), region-based fully convolutional networks (R-FCNs) (Dai et al. 2016), and YOLO (Redmon et al. 2016).

Faster Region-Based Convolutional Neural Networks (Faster R-CNN)

Faster R-CNN performs detection in two stages. Stage 1, called the region proposal network (RPN), involves processing images using a feature extractor (VGG-16), and the class-agnostic box proposals are predicted from the features obtained at some selected intermediate level (conv5).

In Stage 2, features from the same intermediate feature map are extracted using the box proposals and fed to the remainder of the feature extractor to predict the class and the class-

specific box refinement for each proposal. Faster R-CNN is the basis on which most subsequent object detection algorithms, including SSD and R-FCN, were developed.

Single-Shot Multibox Detector (SSD)

SSD architecture is built on VGG-16 architecture. It uses a single feed-forward convolutional network to predict classes and anchor offsets, thereby evading the requirement for a second-stage per-proposal classification operation.

In this approach, the output space of bounding boxes is discretized into a set of default boxes with different object scales and aspect ratios. During prediction, scores for the presence of an object in each default box are generated by the network, and, finally, adjustments are made to the box to match the object shape more accurately.

Region-Based Fully Convolutional Networks (R-FCN)

R-FCN is fundamentally derived from Faster R-CNN, but it is designed to work much faster than Faster R-CNN. In R-FCN, crops are extracted from the last layer of features prior to prediction instead of cropping features from the layer where region proposals are predicted. This minimizes the per-region computation and has been shown to achieve comparable accuracy to Faster R-CNN with less computation time.

Previous research studies have proposed a position-sensitive cropping mechanism in place of the standard region of interest (ROI) pooling operation (Ren et al. 2017). A detailed comparison of these three algorithms (Faster R-CNN, SSD, and R-FCN), along with the speed-accuracy tradeoffs, can be found in a study by Huang et al. (2017).

You Only Look Once (YOLO)

YOLO frames object detection as a regression problem (Redmon et al. 2016). A single neural network is used to predict the bounding boxes and associated class probabilities in a single evaluation over the entire image. Thus, the entire pipeline can be optimized end-to-end based on detection performance. This makes the algorithm very fast, and images can be processed in real-time (45 frames per second [fps]). A detailed description of the YOLO model is provided in Section 3.2 of this report.

Multi-Object Tracking (MOT)

Multi-object tracking (MOT) aims to estimate the states of multiple objects while conserving their identification across time under variations in motion and appearance. This involves determining the locations, velocities, and sizes of the objects across time. With the recent advancements in object detection, tracking-by-detection has emerged as one of the predominant approaches for multi-object tracking. This approach generally involves associating the objects

detected across multiple frames in a video sequence. The two broad categories in a tracking-by-detection framework are batch and online tracking.

Batch methods usually involve determining object trajectories in a global optimization problem and processing the entire video at once. Short tracklets are generated; individual detections are linked first, and then the tracklets are associated globally to form the object trajectory. Flow network formulations (Zhang et al. 2008, Berclaz et al. 2011) and probabilistic graphical models (Yang and Nevatia 2012, Andriyenko et al. 2012) are the two broad classes of algorithms in a batch MOT problem. However, the intensive iterative computation required for generating globally associated tracks and the need for detection of the entire sequence beforehand limits the use of these batch MOT approaches in real-time applications.

Online methods build trajectories sequentially by using information provided up to the present frame and associating the frame-by-frame objects detected. Thus, this approach can be easily implemented for real-time tracking. However, these methods are prone to fragmented trajectory generation under occlusion and object detection errors.

Traditional online MOT methods are multiple hypothesis tracking (MHT) (Reid 1979, Kim et al. 2015) and joint probabilistic data association filter (JPDAF) (Fortmann et al. 1983, Rezatofighi et al. 2015). The JPDAF method involves generating a single state hypothesis by weighting individual measurements with the association likelihoods. MHT, in contrast, involves tracking all possible hypotheses and then applying pruning schemes for computational tractability. Both of these approaches require significant computational and implementation complexity, thereby limiting their implementation in real-time applications.

Recently, Bewley et al. (2016) proposed simple online real-time tracking (SORT), which performs Kalman filtering in the image space and uses the Hungarian algorithm for frame-by-frame data associations. With a state-of-the-art object detection framework (Ren et al. 2017), SORT ranks higher than MHT in the MOT Challenge dataset (Leal-Taixe et al. 2015). However, SORT is known to perform poorly when state estimation uncertainty is high and is known to return substantially high identity switches.

To overcome this shortcoming, Wojke et al. (2017) proposed the Deep-SORT algorithm, which incorporates both motion and appearance information into the association metric, which, in turn, increases robustness against occlusions or detection errors. Even more recently, Bae and Yoon (2018) proposed a robust online MOT method that uses confidence-based data association for handling track fragmentation and deep appearance learning for handling similar object appearance in tracklet association.

In the present study, the research team used YOLOv3 for detecting vehicles, SORT for tracking vehicles and forming trajectories, and semi-supervised trajectory classification for identifying incident trajectories. The experimental results of the proposed approach for incident detection using traffic data obtained from Iowa DOT cameras demonstrate that the framework achieved superior performance compared to supervised learning techniques with a comparable number of

labeled examples. The following chapter provides a detailed analysis and the results of the first research objective, traffic congestion detection from camera images.

CHAPTER 3. TRAFFIC CONGESTION DETECTION FROM CAMERA IMAGES

3.1. Introduction

Traditionally, traffic state estimation is conducted using point-based sensors, including inductive loops, piezoelectric sensors, and magnetic loops (Kotzenmacher et al. 2004). Recent advances in active infrared/laser radar sensors have led to these devices gradually replacing the traditional point-based sensors (Zhong and Liu 2007). Also, with the increasing use of navigation-based GPS devices, probe-based data are emerging as a cost-effective way to collect network-wide traffic data (Feng et al. 2014). Video monitoring and surveillance systems also are used for gathering real-time traffic data (Ozkurt and Camci 2009). Recent advances in image processing techniques have improved the accuracy of vision-based detection. Deep learning methods, such as CNNs, have been able to achieve human-level accuracy in image classification tasks (He et al. 2016). The basic advantage of these methods is that they don't require the identification of hand-crafted features and hence can do away with the painstaking calibration tasks needed when using camera images for traffic state estimation (Bauza et al. 2010).

In this study, camera images taken from different locations and orientations and in different weather conditions were used to successfully detect traffic congestion. Three different models were used for congestion detection tasks. Two of these were deep neural network models: DCNNs and YOLO. Because these models require time-consuming and costly GPU training, SVM, a shallow learning model, was used as a comparison to determine the advantages of using the deep models.

The outline of this chapter is as follows. The present section provides a brief introduction and explains the importance of traffic congestion detection. The next section gives an overview of the proposed models used for traffic congestion determination. The third section provides a description of the data used in this study and the data preprocessing steps adopted for further analyses. The fourth section includes a discussion of the results obtained from the analyses, and the final section provides the conclusions form this study and recommendations for future work. This study was previously published in *Transportation Research Record* (Chakraborty et al. 2018a).

3.2. Methodology

Traffic congestion detection from camera images can be conducted in two broad ways. With the first approach, the input image can be fed into an object recognition model to determine the number of vehicles in the image, and, when the number of vehicles exceeds a certain threshold, the image can be labeled as congested. With the second approach, the entire image can be classified as either congested or non-congested. In this study, the research team used the second approach because it is much simpler and does not require time-consuming manual annotation of individual vehicles.

Three different algorithms for the traffic congestion detection task were tested in this study: two based on deep neural networks, which require time-consuming GPU training, and one from the class of shallow learning algorithms, which does not require GPU training. The shallow algorithm was adopted primarily to determine the advantages, if any, of using a GPU for this classification task. The three algorithms used in this study were as follows:

- Traditional DCNN
- YOLO
- SVM

A detailed description of each of these algorithms is provided in the following sections.

Deep Convolutional Neural Networks (DCCNs)

Collectively, DCNNs are a state-of-the-art technique for object detection and image classification. In this study, we used a traditional CNN architecture consisting of convolution and pooling layers. The convolution architecture used in this study is shown in Table 1.

Table 1. DCNN model architecture used

Layer	Kernel	Stride	Output Shape
Input			[400, 225, 3]
Convolution	3×3	1	[400, 225, 32]
Convolution	3×3	1	[398, 223, 32]
Max Pooling	2×2	2	[199, 111, 32]
Dropout			[199, 111, 32]
Convolution	3×3	1	[199, 111, 64]
Convolution	3×3	1	[197, 109, 64]
Max Pooling	2×2	2	[98, 54, 64]
Dropout			[98, 54, 64]
Dense			512
Dropout			512
Dense			2

Because images from different cameras were used in this study, the input images were of different sizes, the majority being 800×450 pixels. The images were then resized to 400×225 pixels to prevent memory allocation issues during the training of the model. Next, these images were fed into the model as two consecutive convolution layers 32×3×3 in size, followed by a max pooling layer 2×2 in size. This was followed by two additional convolution layers 64×3×3 in size and then again a max pooling layer with a 2×2 filter. Each max pooling layer was followed by a dropout with a probability of 0.25 to prevent overfitting. Finally, two fully connected layers (dense) were used, the first one with 512 neurons and the final one with 2 neurons corresponding to the binary classes (congested and non-congested). A batch size of 32

was used throughout the model, and leaky rectified linear units (ReLU) was used as an activation function.

You Only Look Once (YOLO)

YOLO uses a simple CNN architecture, shown in Table 2.

Table 2. YOLO model architecture used

Layer	Kernel	Stride	Output Shape
Input			[416, 416, 3]
Convolution	3×3	1	[416, 416, 16]
Max Pooling	2×2	2	[208, 208, 16]
Convolution	3×3	1	[208, 208, 32]
Max Pooling	2×2	2	[104, 104, 32]
Convolution	3×3	1	[104, 104, 64]
Max Pooling	2×2	2	[52, 52, 64]
Convolution	3×3	1	[52, 52, 128]
Max Pooling	2×2	2	[26, 26, 128]
Convolution	3×3	1	[26, 26, 256]
Max Pooling	2×2	2	[13, 13, 256]
Convolution	3×3	1	[13, 13, 512]
Max Pooling	2×2	1	[13, 13, 512]
Convolution	3×3	1	[13, 13, 1024]
Convolution	3×3	1	[13, 13, 1024]
Convolution	1×1	1	[13, 13, 35]

This neural network uses only standard layer types: convolution with a 3×3 kernel and max pooling with a 2×2 kernel. The very last convolutional layer has a 1×1 kernel, which serves to reduce the data to the shape 13×13×125. This 13×13 shape is the size of the grid into which the image is divided. There are 35 channels for every grid cell. These 35 channels represent the data for the bounding boxes and the class predictions, because each grid cell predicts five bounding boxes and a bounding box is described by seven data elements:

- x, y, width, and height for the bounding box's rectangle
- Confidence score
- Probability distribution over the two classes (congested and non-congested)

The key implementation steps for YOLO are as follows:

1. The input image is resized to 416×416 pixels.
2. The image is passed through a CNN in a single pass.
3. The CNN outputs a 13×13×k tensor describing the bounding boxes for the grid cells. The value of k is related to the number of classes as follows: k = (number of classes + 5) × 5.

4. The confidence scores for all bounding boxes are computed, and all boxes that fall below a predefined threshold are rejected.

Because there are 13×13 = 169 grid cells and each cell predicts five bounding boxes, there are 845 bounding boxes in total. Ideally, the majority of these boxes would have very low confidence scores. In this study, a confidence threshold of 45% was used for congestion detection.

Support Vector Machine (SVM)

SVM is one of the most widely used shallow algorithms for image classification. It solves a constrained quadratic optimization problem to classify data into different categories. The resulting optimal hyperplane is determined by maximizing the largest minimum distance to training examples to make the hyperplane least sensitive to noise. In this study, the Oriented FAST and Rotated BRIEF (ORB) (Rublee et al. 2011) feature detector was used to detect the key points in each image, whereby the features from accelerated segment test (FAST) algorithm was used to extract the key points, and the Harris corner distance was used to determine the top N points. The algorithm was run on the training dataset with 10-fold cross-validation to determine the optimal penalty parameter and kernel. This algorithm was run on Windows 7 with an Intel Core i7-4790 CPU with 8 GB of RAM.

3.3. Data Description

Two different data sources were used in this study: camera images and radar-based Wavetronix sensors. Camera images were obtained from 121 cameras from the Iowa DOT CCTV camera database spread across the Interstates and highways of Iowa. The database included data from the major cities of Iowa, i.e., Des Moines, Sioux City, Cedar Rapids, Council Bluffs, Davenport, and Iowa City. Images were extracted from the cameras at five-minute intervals from October 2016 through March 2017 (six months), resulting in a total of 3.5 million images during the study period. The task of assigning a label (congested or non-congested) to an image consisted of four sub-tasks:

- Associate each camera with a nearby Wavetronix sensor pair.
- Smoothen the Wavetronix data.
- Extract the details (camera name, timestamp) of each image.
- Assign the label to the image based on sensor data.

The details of each of these tasks are discussed next.

Each camera was first associated with the two nearest Wavetronix sensors covering both directions of the freeway on which the camera was placed. If the sensor pair was located more than 0.5 miles away from the camera, then the particular camera was removed from the analysis. Here, the assumption was made that if sensors are located more than 0.5 miles away from the

camera, then the observation made from the camera might not match up with the sensors' observations.

The next step was to assign the traffic data from the sensor pair to each image. However, the sensor data obtained from Wavetronix in 20-second intervals included too much noise; therefore, we used wavelet smoothing to remove the noise. In this study, among the several families of wavelets that could be used, such as Haar, Daubechies, Biorthogonal, Symlets, Coiflets, Morlet, Mexican Hat, Meyer, etc., Daubechies extremal phase wavelets were used. Daubechies family wavelets are also known as dbN, where N refers to the number of vanishing moments. The higher the value of N, the longer the wavelet filter and the smoother the wavelet. Based on the data, db2 with level 6 was used to achieve a smooth curve-like filter that followed most of the variations in the original signal. A sample of the original and smoothed data is shown in Figure 1.

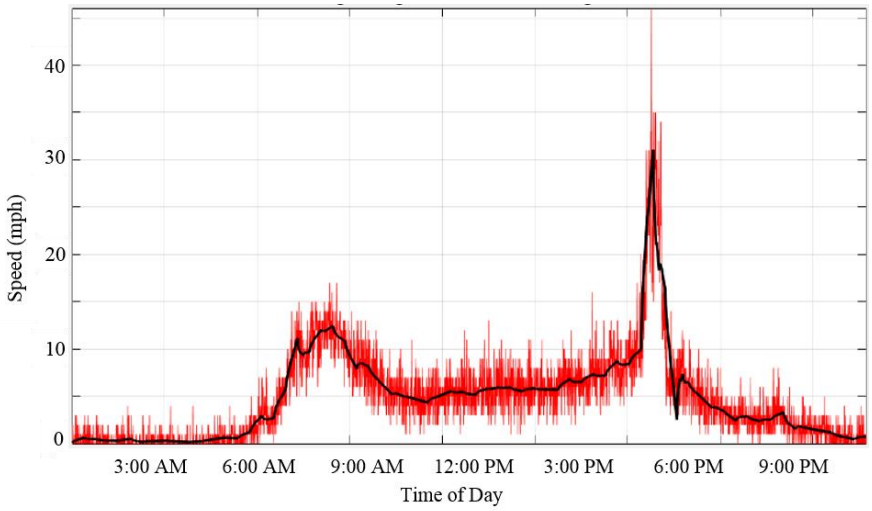


Figure 1. Original and smoothed occupancy using wavelet transform (db2 level 6)

The next step was to extract the details of each image. The top of each image showed the details of the image (direction, camera name, and timestamp). Optical character recognition (OCR) was used to extract the details from each image, which were then matched with the corresponding sensor data based on the camera's name and timestamp.

After obtaining the smoothed Wavetronix data, timestamp, and camera name for each image, the traffic data obtained from the sensors were assigned to the images. The traffic data comprised speed, volume, and occupancy observed at 20-second intervals. To assign a label of congested or non-congested to the images, occupancy values were used, which are denoted by the percentage of the time the sensor is occupied by vehicles and which have a one-to-one mapping to traffic density or the number of vehicles in the unit distance. Persaud and Hall (1989) suggested that an occupancy of 20% or more should be considered congested, whereas an occupancy below that should be considered non-congested. Thus, if no congestion (occupancy < 20%) was observed in either direction of the Wavetronix pair, then the image was classified as non-congested; if congestion was visible in any particular direction or in both directions, it was labeled as

congested. The research team adopted this approach to do away with manual labeling of congested and non-congested images and to follow a uniform methodology for assigning labels to the images.

Finally, 1,218 congested images and more than 3 million non-congested images were obtained. Due to class imbalance, 1,200 non-congested images were randomly chosen out of the 3 million images. This dataset consisting of a total of 2,418 images was then subdivided into a training set and a test set. The training set consisted of 1,400 images with equal proportions of congested and non-congested images. However, the YOLO approach to congestion detection requires manually annotating the region of congestion. For this purpose, 100 congested images were extracted from the training set and manually annotated with the congested region. The test set consisted of 1,018 images, out of which 518 were congested and the rest were uncongested. Because sensor errors can occasionally cause misclassification of images, test set images were manually cross-checked before the final labels were assigned. However, no manual cross-checking of labels was performed for the training set because it was assumed that the algorithm itself should be able to determine the misclassifications, if any, in the training set.

3.4. Results

The performance of each of the three algorithms was trained on 1,400 training set images and tested on 1,018 test set images (518 congested and 500 non-congested). YOLO was trained and tested on an NVIDIA GTX 1080 Ti GPU with 8 GB of RAM, while DCNN was trained and tested on an NVIDIA Tesla K20m GPU with 4 GB of RAM. An Intel Core i7-4790 CPU with 8 GB of RAM was used for training and testing SVM. The training times for YOLO, DCNN, and SVM were 22 hours, 26 minutes, and 50.4 seconds, respectively. The testing times for the three algorithms were 0.01, 0.01, and 0.03 seconds/frame, respectively. The testing time did not include the time required for developing the model; rather, it included only the time required to predict the class for each image. Because YOLO and DCNN are deep models, they had to be trained and tested using GPUs, which involved time-consuming and costly training compared to their shallow counterpart, SVM. The testing times for DCNN and YOLO were lower than for SVM, but they required that a GPU be used for testing.

The performance of the algorithms was evaluated using the standard performance metrics of precision, recall, and accuracy (Equations 1, 2, and 3, respectively).

$$Precision = \frac{TP}{TP+FP} \quad (1)$$

$$Recall = \frac{TP}{TP+FN} \quad (2)$$

$$Accuracy = \frac{TP+TN}{TP+FP+TN+FN} \quad (3)$$

When a congested image was correctly labeled (i.e., the predicted label was also congested), the result was classified as true positive (TP). Similarly, if a non-congested image was correctly

labeled as non-congested, the result was classified as true negative (TN). However, if the actual label was congested and the predicted label was non-congested, the result was classified as false negative (FN). Finally, if the actual label was non-congested and the predicted label was congested, the result was classified as false positive (FP).

Some examples of true classifications and misclassifications obtained from each algorithm (YOLO, DCNN, and SVM) are shown in Figure 2.

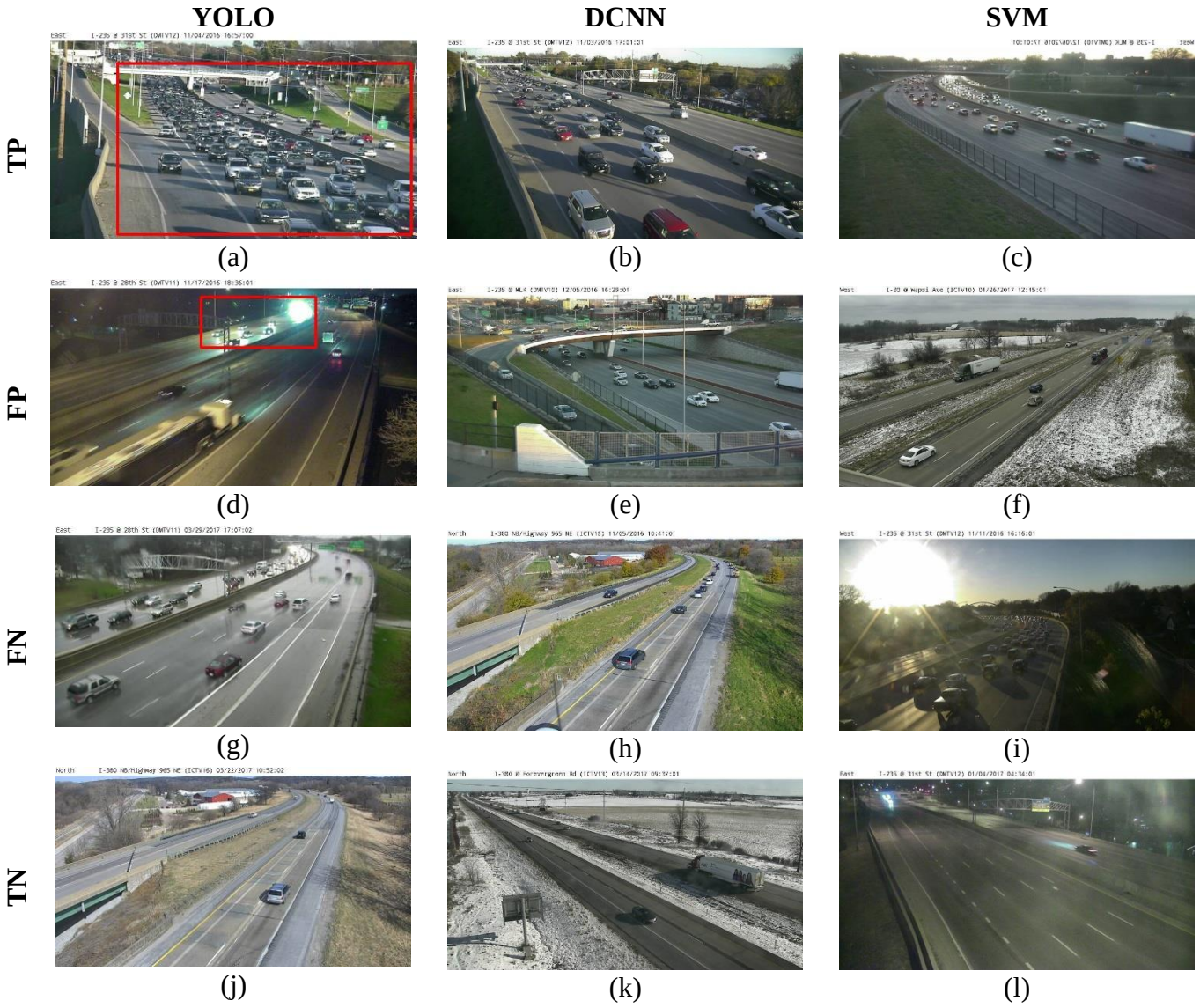


Figure 2. Congestion detection classification examples: (a-c) true positives, (d-f) false positives, (g-h) false negatives, (j-l) true negatives

Examples of true positives, in which each algorithm correctly labeled congested images, are shown in Figure 2(a) through (c) (YOLO also gives the bounding box for the congested region). Examples of false positives, in which the algorithms misclassified non-congested images as congested, are shown in Figure 2(d) through (f). It can be seen that YOLO misclassified an image as a congested region because of a group of vehicles located far away from the camera during nighttime (Figure 2[d]), and vehicles on a bridge led to misclassification by DCNN (Figure 2[e]). SVM had misclassifications in adverse weather conditions (Figure 2[f]) because snow particles were detected as corners, which caused the image to be labeled as congested. Examples of false negatives, in which the algorithms failed to detect congested images correctly, are shown in Figure 2(g) through (i). Congestion quite distant from the camera led to misclassification by YOLO (Figure 2[g]), whereas DCNN failed to detect congestion in a single lane when the other lane was closed and hence empty (Figure 2[h]). Glare issues resulted in SVM misclassifications (Figure 2[i]). Finally, examples of true negatives, in which the algorithms correctly labeled non-congested images, are shown in Figure 2(j) through (l).

The precision, recall, and accuracy values obtained for each algorithm are shown in Table 3.

Table 3. Precision, recall, and accuracy values obtained for the three algorithms

Method	Precision (%)	Recall (%)	Accuracy (%)
YOLO	88.6	94.3	91.4
DCNN	86.9	93.9	90.2
SVM	82.8	88.5	85.7

YOLO achieved the highest precision, recall, and accuracy, followed closely by DCNN. Because YOLO achieved better accuracy compared to DCNN, we did not perform region-based CNN separately to determine the congested region of the results obtained from DCNN. YOLO, however, being a region-based classifier, gives the congested region of the image by default (see Figure 2[a] and Figure 2[d]). The accuracy obtained by SVM was comparatively lower (85.2%) than expected given the lower computation costs involved in such a shallow algorithm. In this context, it should be mentioned that a separate analysis using an ensemble of shallow learning algorithms (SVM with ORB, Shi-Tomasi, and Structured Edge Toolbox feature detectors) yielded an accuracy of 86.7% with the same dataset. In this study, however, we used only SVM with ORB to compare deep learning models to a standard shallow model.

Sensitivity Analysis

Sensitivity analysis was also performed to determine which factors might affect the performance of the congestion detection system developed in this study. The two factors that could influence the classification task were evaluated: first, the time of day the image was captured (daytime versus nighttime) and, second, camera resolution (blurring, rain, snow, and glare). The test database was then divided into four subgroups according to the combination of the two factors, as follows:

- D-G: daytime, good resolution (436 images)

- N-G: nighttime, good resolution (147 images)
- D-P: daytime, poor resolution (190 images)
- N-P: nighttime, poor resolution (245 images)

Receiver operating characteristics (ROC) curves were then used to compare the performance of each algorithm for each subgroup based on the true positive rate (TPR) and false positive rate (FPR), as defined in Equations (4) and (5), respectively:

$$TPR = \frac{TP}{TP+FN} \tag{4}$$

$$FPR = \frac{FP}{FP+TN} \tag{5}$$

For an efficient image classification model, the TPR should be higher than the corresponding FPR. However, for a system model with poor vision, when the sensitivity (TPR) increases, the model loses the ability to discriminate between congested and non-congested images, which makes the TPR directly proportional to the FPR. The ROC curves for each subgroup obtained from the three models—YOLO, DCNN, and SVM—are shown in Figure 3(a) through (c). The overall ROC curve for the three algorithms is shown in Figure 3(d). The area under each curve (AUC) is also provided for each case.

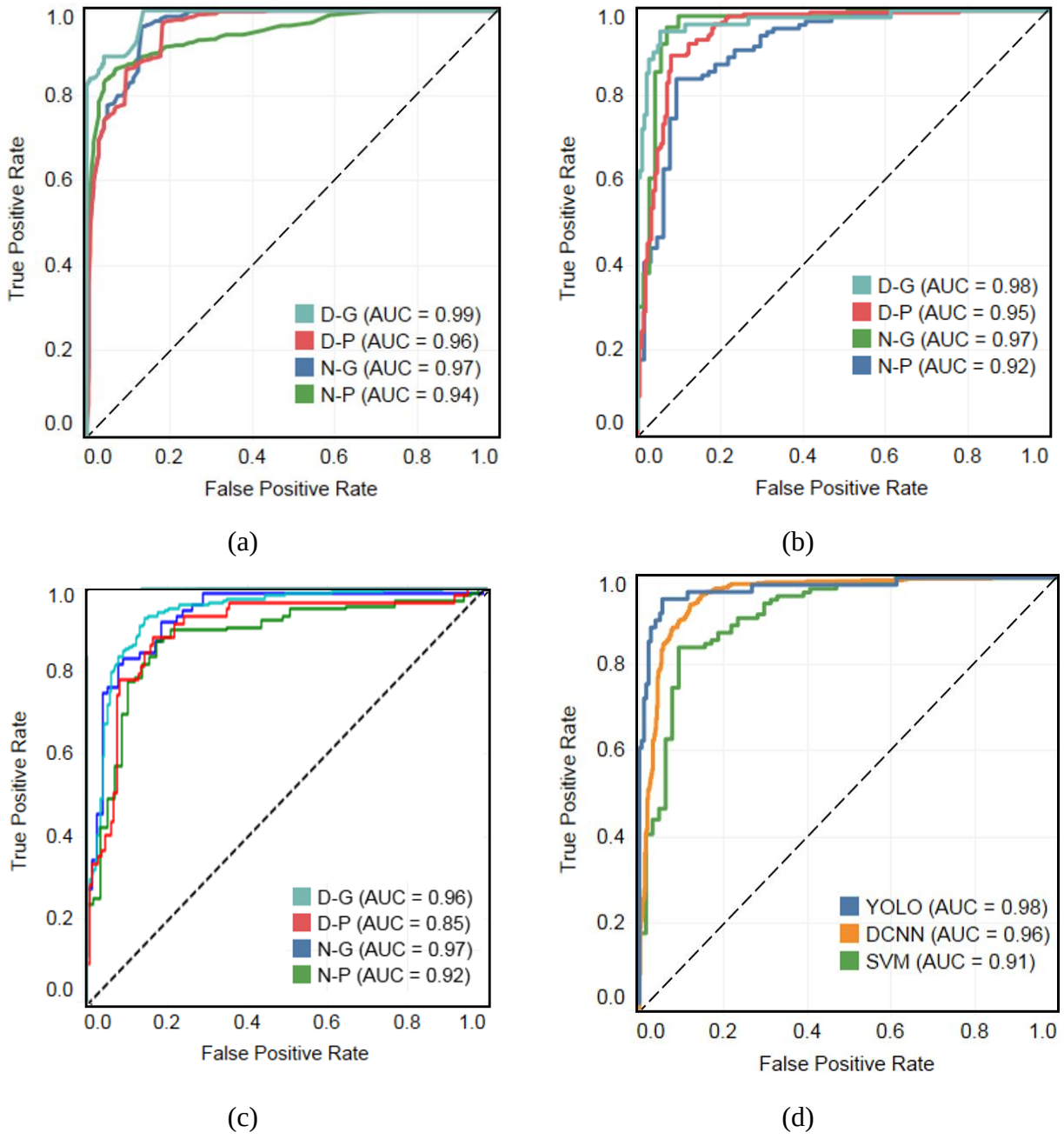


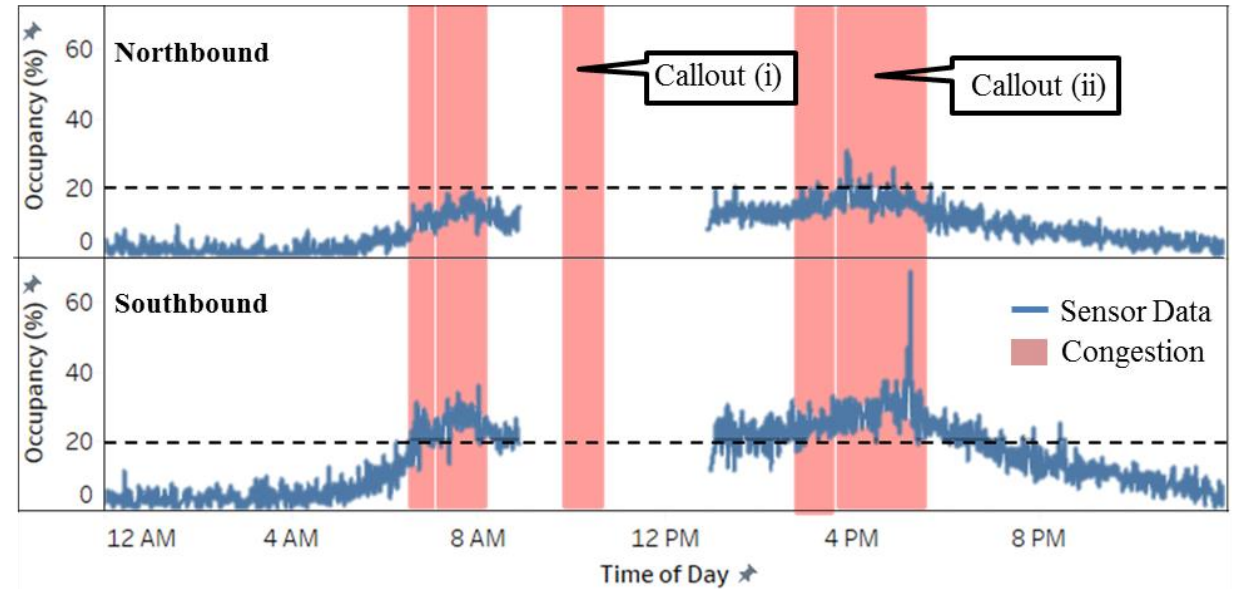
Figure 3. ROC curves under different prevalent conditions obtained from (a) YOLO, (b) DCNN, (c) SVM, and (d) all conditions combined for each algorithm

For all three algorithms, the TPRs were higher than the corresponding FPRs irrespective of the prevailing conditions (daytime or nighttime, poor or good resolution). All of the algorithms performed well during the daytime, irrespective of the camera resolution. However, the AUCs were found to be lowest for poor-resolution images at night (N-P). Moreover, irrespective of the conditions, the AUCs from all algorithms for each subgroup were found to be mostly higher than 0.90, except for the case of SVM under N-P conditions. This shows that the system works well even under challenging conditions.

In addition, ROC curves can be used by traffic management centers (TMCs) to choose an optimal threshold between TPR and FPR. Previous studies have shown that too many false calls are a major reason for the limited integration of automatic incident detection algorithms in TMCs. Hence, it is important for TMC personnel to know the accuracy that can be achieved with an algorithm given a particular FPR. For example, if a TMC wants to restrict the FPR to lower than 0.1, then the TPRs obtained by YOLO, DCNN, and SVM must be 0.92, 0.96, and 0.82, respectively, during good daytime (D-G) conditions. Obviously, the accuracy would be lower under poor camera conditions at night. TMC personnel can use the ROC curves to set optimal TPR and FPR thresholds based on their specific needs.

Real-Time Implementation

The congestion detection algorithms developed in this study can also be implemented online easily. With a test time of 0.01 seconds per image, the algorithms can be used to detect traffic congestion using approximately 1,000 cameras at an interval of every 10 seconds using a single GPU. Figure 4 shows an example of congestion detection by the DCNN algorithm on images extracted from a camera on a single day (October 27, 2017) at an interval of every 10 seconds.



(a)

(b)

(c)

Figure 4. (a) Sensor occupancy data and congestion alerts from a camera on a particular date; (b-c) camera images of Callouts (i) and (ii) identified in (a)

The congestion alerts from the camera are highlighted in Figure 4(a), along with the occupancy data obtained from nearest radar sensors in both directions of traffic. However, due to sensor issues, sensor data were missing from 8:51 a.m. to 12:57 p.m. Therefore, a two-vehicle crash reported at around 10:30 a.m. was missed by the sensor but was detected successfully by the camera. This example also shows that using multiple data sources (cameras, sensors, etc.) can increase the reliability of traffic state estimation. Camera images corresponding to Callouts (i) and (ii) in Figure 4(a) are provided in Figure 4(b) and (c) to show samples of camera images when congestion alerts are triggered. To eliminate false alerts, alerts are triggered only when congestion is detected on three consecutive frames at 10-second intervals (i.e., persistency test). Also, multiple alerts triggered within 5 minutes of each other are combined together to form a single continuous alert. These “signal smoothing” techniques help decrease false alert rates (FARs) and increase detection rates (DRs). Future studies can be done that implement better smoothing techniques like Fourier transforms or wavelet smoothing to determine the DR and FAR on a network of cameras.

3.5. Conclusions

Recent advancements in machine vision algorithms and high-performance computing have improved image classification accuracy to a great extent. In this study, two such deep learning techniques, the traditional DCNN and YOLO models, were used to detect traffic congestion from camera images. SVM also was used for comparison and to determine the improvements that might be obtained using costly GPU techniques. To eliminate the time-consuming task of manual labeling and to maintain uniformity in congestion labeling, Wavetronix sensors near each camera were used to correctly identify congested images. For testing purposes, each image was also labeled manually to remove misclassifications due to sensor errors.

The YOLO model achieved the highest accuracy of 91.2%, followed by DCNN with an accuracy of 90.2%; 85% of images were correctly classified by SVM. Congestion regions located far away from the camera, single-lane blockages, and glare issues were found to affect the accuracy of the models. To determine the sensitivity of the models to different camera configurations and light conditions, ROC curves were used. All of the algorithms were found to perform well in daytime conditions, but nighttime conditions were found to affect the accuracy of the vision system. However, for all conditions, the AUCs were found to be greater than 0.9 for the deep models. This result shows that the models perform well in challenging conditions as well.

An example of the real-time implementation of congestion detection using the DCNN algorithm was also performed using a continuous set of images extracted from a camera. Simple persistence test methods were applied to reduce false alerts and smoothen the output signal. Future studies can look into different smoothing techniques (e.g., Fourier transform or wavelets) to denoise the output obtained from the algorithm and determine the overall detection rate and false alert rate on a network of cameras. Future studies can also be done using different model architectural designs to improve detection accuracies. Such models can also be used to determine different levels of congestion (high, medium, or low) and to more accurately determine traffic state (speed, volume, and occupancy). The congestion status obtained from the cameras can also be stored as historical data and used to determine traffic anomalies such as incidents.

CHAPTER 4. SEMI-SUPERVISED LEARNING APPROACH FOR FREEWAY INCIDENT DETECTION FROM VIDEOS

4.1. Introduction

Approaches to traffic incident detection from CCTV cameras can be broadly classified into two categories: explicit event recognition by supervised learning and unsupervised learning based on anomaly detection. While supervised techniques can, in general, provide better results in detection or classification tasks, the main hindrance in their application is the scarcity of supervised data samples and the cost of manually annotating and labeling the dataset. In particular, manually annotating vehicle tracks in a video stream is extremely labor-intensive, expensive, and not scalable.

In this study, the research team established a new learning framework for traffic incident detection using recent advances in semi-supervised learning (Loog 2016). Via this framework, the "best of both worlds" can be achieved; only a small sample of normal vehicle tracks and the tracks of vehicles involved in an incident were manually annotated, and all other (unlabeled) vehicle tracks were used to improve the performance of the classification. Experimental results using traffic data collected from Iowa DOT cameras demonstrate that the framework can achieve superior performance compared to supervised learning techniques with a comparable number of labeled examples. This study was previously published in the proceedings of the 21st IEEE Conference on Intelligent Transportation Systems (Chakraborty et al. 2018b).

4.2. Methodology

Traffic incident detection from videos using trajectory information comprises three basic tasks: vehicle detection, vehicle tracking and trajectory formation, and trajectory classification. Each task is described in the following sections, with the primary focus of this study being trajectory classification using semi-supervised techniques.

Vehicle Detection

In recent years, the evolution of CNNs has resulted in significant improvements in the performance of object detection and classification. Various state-of-the-art object detection algorithms developed over the past few years have been based on CNN, including R-CNN (Girshick et al. 2014), Faster R-CNN (Ren et al. 2017), Mask R-CNN (He et al. 2017), deformable convolutional networks (ConvNets) (Dai et al. 2017), SSD (Liu et al. 2016), YOLO (Redmon et al. 2016), YOLOv2 (Redmon and Farhadi 2017), and YOLOv3 (Redmon and Farhadi 2018).

In this study, the research team chose YOLOv3 (Redmon and Farhadi 2018) for vehicle detection primarily because of its fast performance with reasonable accuracy, which makes it suitable for real-time applications. Current object detection systems repurpose powerful CNN classifiers to perform detection. For example, to detect an object, these systems take a classifier

for that object and evaluate it at various locations and scales in the test image. In contrast, YOLO reframes object detection: instead of looking at a single image a thousand times to detect an object, YOLO only looks at an image once (but in a clever way) to perform the full detection pipeline (see Figure 5).

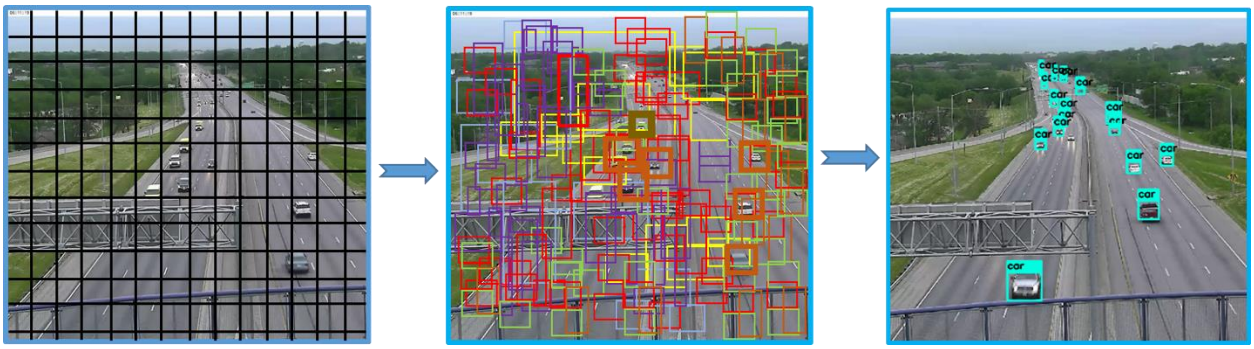


Figure 5. Confidence score prediction of bounding boxes by YOLO, with colors and bounding box widths indicating confidence score probabilities

A single convolutional network simultaneously predicts multiple bounding boxes and class probabilities for those boxes. This makes YOLO extremely fast and easy to generalize to different scenes.

In this study, the YOLOv3-416 model trained on the Microsoft Common Objects in Context (COCO) dataset (Lin et al. 2014) was used for vehicle detection. Out of the 80 classes in the COCO dataset, the car, motorbike, bus, and truck classes were used for the vehicle detection module.

Vehicle Tracking and Trajectory Formation

Recent improvements in the performance of object detection have led tracking-by-detection to become the leading paradigm for MOT. In MOT, multiple objects are detected in each frame, with the aim being to associate the detections across frames in a video sequence. The data association can be performed in batch (Kim et al. 2015, Rezatofighi et al. 2015) or online (Bewley et al. n.d. Wojke et al. 2017).

In this study, the SORT algorithm for vehicle tracking (Bewley et al., n.d.) was used. This is an online multi-object tracking algorithm that uses the Kalman filter and the Hungarian algorithm to address the data association problem. This tracker was chosen because of its reasonable performance in online, real-time settings. SORT updates at 260 Hz, making it suitable for real-time implementation.

The object tracker module outputs a sequence of bounding box coordinates, X-center (X^c) and Y-center (Y^c), for each unique vehicle ID across the frames, thereby forming a trajectory. Thus, a trajectory can be defined as a sequence of two-dimensional points, denoted as

$TR_i = (p_1, p_2, p_3 \ldots p_j \ldots p_{len_i})$. Here, each p_j is a two-dimensional point representing the bounding box coordinates. The length len_i of a trajectory can be different for different trajectories. Note that in this study only the bounding box center coordinates were used, but other features, such as bounding box appearance descriptors, can also be included.

Semi-Supervised Trajectory Classification

The aim of semi-supervised learning is to exploit easily available unlabeled data to improve the performance of supervised classifiers. However, it is not always the case that the semi-supervised classifiers achieve lower error rates compared to their supervised counterparts. On the contrary, empirical studies have observed severely deteriorated performance for semi-supervised classifiers (Ben-David et al. 2008). Recently, Loog (2016) demonstrated how maximum likelihood (ML) can be used to improve classification performance in a semi-supervised setting.

In this study, the problem of trajectory classification in a semi-supervised setting was addressed using contrastive pessimistic likelihood estimation (CPLE) based on ML estimation (Loog 2016). The details of the CPLE method for semi-supervised classification is discussed in the following section, followed by a brief description of the traditional algorithms that were chosen for comparison.

Contrastive Pessimistic Likelihood Estimation (CPLE)

The two main concepts that form the core of CPLE are contrast and pessimism. The CPLE method is contrastive, meaning that the objective function explicitly controls the potential improvements of the semi-supervised classification over the supervised counterpart. CPLE is also pessimistic, which means that the unlabeled data are modeled to behave adversarially so that any semi-supervised learning mechanism benefits least from the unlabeled data. This makes CPLE resilient to whatever form the true (unobserved) labels of the unlabeled data take.

For a *K*-class supervised classification, the log-likelihood objective function is given by

$$L(\theta \mid X) = \sum_{i=1}^{N} \log p(x_i y_i \mid \theta) = \sum_{k=1}^{K} \sum_{j=1}^{N_k} \log p(x_{ij}, k \mid \theta) \tag{6}$$

where, class k contains N_k samples, $N = \sum_{k=K}^{N_k}$ is the total samples, $X = \{(x_i, y_i)\}_{i=1}^{N}$ is the set of labeled training pairs with $x_i \in \Re^d$ being the d-dimensional feature vectors, and $y_i \in C = \{1, ..., K\}$ are their corresponding labels.

The supervised ML estimate, $\hat{\theta}_{\text{sup}}$, maximizes the above criterion:

$$\hat{\theta}_{\text{sup}} = \arg\max_{\theta} L\left(\theta \mid X\right) \tag{7}$$

In this study, linear discriminant analysis (LDA) was chosen as the classifier, similar to the approach of Loog (2016). Here, the log-likelihood objective function is given by

$$\begin{aligned} L_{LDA}\left(\theta \mid X\right) &= \sum_{i=1}^{N} \log p\left(x_i, y_i \mid \pi_1, ..., \pi_k, \mu_1, ..., \mu_k, \Sigma\right) \\ &= \sum_{k=1}^{K} \sum_{j=1}^{N_k} \log p\left(x_{kj}, k \mid \pi_k, \mu_k, \Sigma\right) \end{aligned} \tag{8}$$

where, $\theta = \left(\pi_1, ..., \pi_k, \mu_1, ..., \mu_k, \Sigma\right)$, π_k are the class priors, μ_k are the class means, and Σ is the class conditional covariance matrix.

Let us define the fully labeled dataset by

$$X_V = X \cup \left\{\left(u_i, v_i\right)\right\}_{i=1}^{M} \tag{9}$$

Then, $\hat{\theta}_{\text{opt}}$ gives the parameter estimates of the classifier where the unlabeled data are also labeled.

$$\hat{\theta}_{\text{opt}} = \arg\max_{\theta} L\left(\theta \mid X_V\right) \tag{10}$$

Since supervised parameters in $\hat{\theta}_{\text{sup}}$ are estimated on a subset X of X_V, we have

$$L\left(\hat{\theta}_{\text{sup}} \mid X_V\right) \leq L\left(\hat{\theta}_{\text{opt}} \mid X_V\right) \tag{11}$$

In semi-supervised setting, V is unobserved, but the labeled and unlabeled data (X and U) are available. The semi-supervised setting has more information compared to a supervised setting but less than the fully labeled case. Thus,

$$L\left(\hat{\theta}_{\text{sup}} \mid X_V\right) \leq L\left(\hat{\theta}_{\text{semi}} \mid X_V\right) \leq L\left(\hat{\theta}_{\text{opt}} \mid X_V\right) \tag{12}$$

Now, the supervised estimate is taken into account explicitly in order to construct a semi-supervised classifier that can improve upon its supervised counterpart.

Before doing so, q_{ki} is defined to be the hypothetical posterior of observing label k given feature vector u_i. It can be also interpreted as the soft label for u_i. Since $\sum_{k \in C} q_{ki} = 1$, the K-dimensional vector $q_{\cdot i}$ can be stated as an element of the simplex Δ_{K-1} in $\Re^K$:

$$q_{\cdot i} \in \Delta_{K-1} = \left\{ (\rho_1, \ldots, \rho_K)^T \in \Re^K \mid \sum_{i=1}^{K} \rho_i = 1, \rho_i \geq 0 \right\} \tag{13}$$

Provided that the posterior probabilities are defined, the log-likelihood on the complete dataset for any parameter vector θ can be expressed as

$$L(\theta \mid X, U, q) = L(\theta \mid X) + \sum_{i=1}^{M} \sum_{k=1}^{K} q_{ki} \log p(u_i, k \mid \theta) \tag{14}$$

where the variable *q* on the left-hand side explicitly indicates the dependence on q_{ki}.

The relative improvement of the semi-supervised estimate θ over the supervised solution for a given *q* can be expressed as

$$CL\left(\theta, \hat{\theta}_{\text{sup}} \mid X, U, q\right) = L(\theta \mid X, U, q) - L\left(\hat{\theta}_{\text{sup}} \mid X, U, q\right) \tag{15}$$

This enables the extent of improvement of the semi-supervised estimates to be checked in terms of log-likelihood, defined as contrast. Since *q* is unknown, the most pessimistic solution can be chosen where it is assumed that the true (soft) labels achieve the worst case among all of the semi-supervised solutions and *q* is chosen such that the likelihood gain is minimized. Thus, the objective function can be written as

$$CPL\left(\theta, \hat{\theta}_{\text{sup}} \mid X, U\right) = \min_{q \in \Delta_{K-1}^{M}} CL\left(\theta, \hat{\theta}_{\text{sup}} \mid X, U, q\right) \tag{16}$$

where $\Delta_{K-1}^{M} = \prod_{i=1}^{M} \Delta_{K-1}$ is the Cartesian product of *M* simplices.

The objective function is strictly concave in θ and linear in *q*. The heuristic to solve the maximization problem is based on alternating between the following two steps:

Given a soft labeling *q*, the optimal LDA parameters are estimated by

$$\hat{\pi}_k = \frac{N_k + \sum_{i=1}^{M} q_{ki}}{N + M} \tag{17}$$

$$\hat{\mu}_k = \frac{\sum_{j=1}^{N_k} x_{kj} + \sum_{i=1}^{M} q_{ki} u_i}{N_k + \sum_{i=1}^{M} q_{ki}} \tag{18}$$

$$\Sigma = \frac{1}{N+M} \sum_{k=1}^{K} \left[\sum_{j=1}^{N_k} \left(x_{kj} - \hat{\mu}_k \right) \left(x_{kj} - \hat{\mu}_k \right)^T + \sum_{i=1}^{M} q_{ki} \left(u_i - \hat{\mu}_k \right) \left(u_i - \hat{\mu}_k \right)^T \right] \tag{19}$$

The gradient ∇ for q given θ is calculated, and q is changed to $q - \alpha\nabla$, with step size $\alpha > 0$. The step size α is decreased as one over the number of iterations, and the maximum number of iterations is restricted to 3,000.

Baseline Algorithms

The performance of the above CPLE-based framework for trajectory classification was compared to that of two baseline semi-supervised methods: self learning (Zhu and Goldberg 2009) and label spreading (Zhou et al. 2004). Self learning combines information from unlabeled and labeled data to iteratively identify the labels for the unlabeled data. The labeled training set is enlarged on each iteration until the entire dataset is labeled. LDA (Balakrishnama and Ganapathiraju 1998) was used as the base model for self learning in the present study. Label spreading (Zhou et al. 2004), a modification of the traditional label propagation algorithm (Zhu and Ghahramani 2002), uses an affinity matrix based on normalized graph Laplacian. It uses soft clamping for labeling, and the loss function has regularization properties that make it robust to noise. Interested readers can refer to Bengio et al. (2006) for further details. Besides these two baseline algorithms, the results from the CPLE-based framework were also compared to those from its supervised counterpart obtained from the LDA classifiers trained on the labeled data.

Feature Vector Generation

The trajectories obtained from the vehicle tracker module are of variable length. However, the semi-supervised techniques described above require fixed-dimensional feature vectors. Hence, trajectory subsampling was first used to convert these variable-length trajectories to fixed-length trajectories, similar to Piciarelli et al. (2008). Each trajectory was subsampled to form a list of two-dimensional coordinates. Since the typical length of each trajectory was from 70 to 80, the research team heuristically chose 75 as the fixed length of each of these lists. Thus, each trajectory was defined as $TR_i = p_1 p_2 p_3 \ldots p_j \ldots p_{75}$, where p_j is the two-dimensional vector representing $\left[X_j^c, Y_j^c \right]$. The feature vectors were normalized to zero mean, and principal component analysis was performed for dimension reduction. The research team found that 95% of the variance (explained by the top three principal components each for X^c and Y^c) is sufficient, similar to Loog (2016). Finally, the top three principal components for X^c and Y^c were concatenated to form a six-dimensional vector representing the trajectory information for each vehicle ID. This six-dimensional feature vector was used for trajectory classification.

4.3. Data Description

The primary source of data used in this study was the traffic incident videos obtained from the CCTV cameras installed by the Iowa DOT along the freeways of Iowa. The dataset consisted of 151 traffic incident videos recorded from these cameras during the period from January 2016 through December 2017 (24 months). Each video was of two-minutes in duration and was recorded at 30 frames per second, and each video clearly captured the onset of the traffic incident. The resolution of the videos varied from 800×480 pixels to 1920×1080 pixels, depending on the camera resolution. The traffic incidents were caused by car crashes or stalled vehicles. Of the 151 incident videos, 11 videos were manually annotated with the bounding boxes of the vehicles involved in the incident. A JavaScript-based video annotation tool (Bolkensteyn 2016) based on VATIC (Vondrick et al. 2013) was used for annotating the vehicles. This annotation resulted in a total of 15 unique trajectories of vehicles involved in incidents. These trajectories were then matched with the vehicle trajectories obtained from the object detection and tracking modules used in this study (YOLOv3 for vehicle detection and SORT for vehicle tracking). For each video frame, each manually annotated bounding box was matched with the detected bounding box with maximum overlap, with a minimum threshold of 0.5 Intersection over Union (IoU). Each manually annotated incident trajectory was successfully matched with a unique trajectory obtained from the tracking algorithm. These trajectories are henceforth referred to as incident trajectories. The remaining trajectories in the 11 manually annotated incident videos were classified as normal trajectories. Fifteen such normal trajectories were randomly selected into the labeled dataset. Thus, the labeled dataset consisted of 15 normal trajectories and 15 incident trajectories.

Ninety incident videos were randomly selected from the 151 incident videos to prepare the unlabeled dataset. The 11,685 trajectories obtained by the object detection and tracking algorithm from those 90 videos were included in the unlabeled dataset. The remaining 50 incident videos were equally divided into validation and test datasets, with 25 incident videos in each set. Fifty baseline videos without any incidents were also randomly selected and divided equally into validation and test datasets. Thus, the validation and test datasets consisted of 50 videos each, 25 of them being incident videos and the remaining 25 being normal baseline videos. The validation and test datasets consisted of 6,333 and 5,375 trajectories, respectively.

4.4. Results

The research team used the state-of-the-art object detection algorithm YOLOv3 for vehicle detection and SORT for vehicle tracking. The object detection and tracking process ran at around 55 fps on an NVIDIA GTX 1080 Ti GPU, making it suitable for real-time performance. Figure 6 shows sample images of the vehicles detected.

Figure 6. Sample images of vehicle detections

A sample image of the results of vehicle tracking is shown in Figure 7, where each color represents a unique trajectory.

Figure 7. Sample vehicle tracking results

The labeled trajectory dataset consisted of 15 incident trajectories and 15 normal trajectories. To determine the sensitivity of the algorithm to the number of labeled examples, each algorithm was tested for labeled sample sizes varying from 5 to 15 trajectories for each class (normal and incident). The efficacy of the proposed model (CPLE) along with that of the comparison models (label spreading, self learning, and supervised learning) were validated using the validation dataset, and the final accuracy was reported for the test dataset. To recall, the validation and test datasets consisted of 50 videos each, 25 of them being incident videos and remaining 25 being non-incident/baseline videos.

A video was labelled as an incident video if at least one trajectory in the video was classified as an incident trajectory by the algorithm. The accuracy of the algorithm (ACC) is given by the number of correct classifications of incident videos (TPR) and baseline videos (TNR), as shown in Equations 20 through 22.

$$TPR = \frac{TP}{P} \tag{20}$$

$$TNR = \frac{TN}{N} \tag{21}$$

$$ACC = \frac{TP + TN}{P + N} \tag{22}$$

TP and TN refer to the respective numbers of correctly identified incident and baseline videos, while P and N refer to the respective total numbers of incident and baseline videos (25 each).

Figure 8 shows the accuracy of each algorithm on the validation dataset for different numbers of labeled samples.

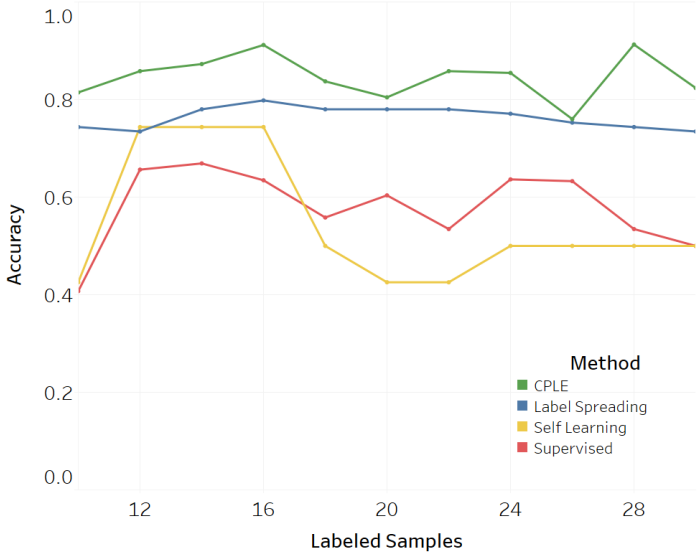


Figure 8. Accuracy of the algorithms for different numbers of labeled samples

The experiments were repeated 20 times, and the average accuracy of each algorithm was reported. It can be clearly seen in Figure 8 that CPLE exhibited superior performance compared to the other semi-supervised approaches and its supervised counterpart. On average, a 14% improvement was obtained when using CPLE compared to the second best algorithm (label spreading). The best model obtained from each algorithm was selected and applied to the test dataset. Table 4 shows the accuracy of each algorithm on the test dataset.

Table 4. Accuracy of the algorithms on the test dataset

Method	TPR	TNR	ACC
CPLE	0.83	0.92	0.88
Label Spreading	0.60	0.94	0.77
Self Learning	0.53	0.88	0.71
Supervised	0.28	0.96	0.62

Table 4 shows that while CPLE successfully identified a large majority of the incident videos (21 of the 25 incident videos), the other algorithms failed to do so and performed poorly in terms of

TPR. However, since the majority of trajectories were normal trajectories, all algorithms performed well in correctly identifying the baseline videos. This result shows that the CPLE algorithm successfully extracted information regarding both incident and normal trajectories from the unlabeled dataset and hence achieved better performance than the other algorithms.

Figure 9 shows a sample of incident and normal trajectories labeled by the CPLE algorithm for three incident videos (Video IDs 1, 2, and 3).

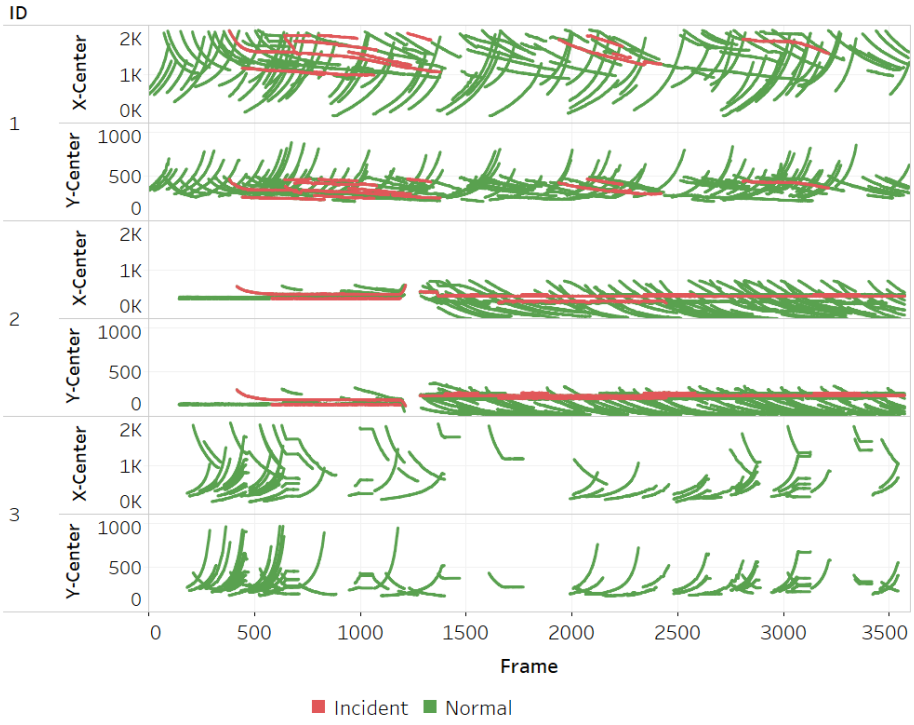


Figure 9. Incident and normal trajectories labeled by the CPLE algorithm for three incident videos

The x and y coordinates of the bounding box center of each vehicle across the video frames are shown in the figure. The CPLE algorithm successfully detected the incident trajectories in Video IDs 1 and 2 but failed to detect any incident trajectory in Video ID 3, primarily due to failed object detection caused by poor video quality. An example of a stalled vehicle detected across three frames is shown in Figure 10.

Figure 10. Sample images of stalled vehicle detected across three frames taken at one-second intervals

4.5. Conclusions

State DOTs typically install a large number of CCTV cameras across freeways for surveillance purposes. However, it is virtually impossible to manually monitor such a large network of cameras constantly. Hence, there is a significant need to develop automatic incident detection algorithms that use the data from these cameras.

Incident detection from cameras has typically been approached using either supervised or unsupervised algorithms. A major hindrance in the application of supervised techniques for incident detection is the lack of a sufficient number of incident videos and the labor-intensive, costly annotation tasks involved in the preparation of a labeled dataset.

In this study, the research team approached the incident detection problem using semi-supervised techniques. Maximum likelihood estimation-based contrastive pessimistic likelihood estimation (CPLE) was used for trajectory classification and identification of incident trajectories. Vehicle detection was performed using state-of-the-art deep learning-based YOLOv3, and SORT was used for tracking. Results showed that CPLE-based trajectory classification outperformed the traditional semi-supervised techniques (self learning and label spreading) and its supervised counterpart by a significant margin.

CHAPTER 5. CONCLUSION

Automatic traffic anomaly detection has been identified to be crucial for the reduction of non-recurrent congestion caused by incidents. In this study, the research team proposed an anomaly detection framework utilizing images and videos from CCTV cameras. State DOTs typically install a large number of CCTV cameras across freeways for surveillance purposes. However, it is virtually impossible to manually monitor such a large network of cameras constantly. Hence, there is a significant need to develop automatic incident detection algorithms that use the data from these cameras.

This study was divided into two broad topics involving the detection of freeway traffic anomalies from cameras. The first research objective involved detecting traffic congestion from camera images. Two modern deep learning techniques, the traditional DCNN and YOLO models, were used to detect traffic congestion from camera images. The SVM model also was used for comparison and to determine the improvements that might be obtained using costly GPU techniques.

To eliminate the time-consuming task of manual labeling and to maintain uniformity in congestion labeling, the research team used nearby Wavetronix sensors to correctly identify congested images. For testing purposes, each image was labeled manually to remove misclassifications due to sensor errors.

The YOLO model achieved the highest accuracy of 91.2%, followed by the DCNN model with an accuracy of 90.2%; 85% of images were correctly classified using the SVM model. Congestion regions located far away from the camera, single-lane blockages, and glare issues were found to affect the accuracy of the models.

To determine the sensitivity of the models to different camera configurations and light conditions, ROC curves were used. All of the algorithms were found to perform well in daytime conditions, but nighttime conditions were found to affect the accuracy of the vision system. However, for all conditions, the AUCs were found to be greater than 0.9 for the deep models. This result shows that the models performed well in challenging conditions as well.

The second part of this study aimed at detecting traffic incidents from CCTV videos. Incident detection from cameras has typically been approached using either supervised or unsupervised algorithms. A major hindrance in the application of supervised techniques for incident detection is the lack of a sufficient number of incident videos and the labor-intensive, costly annotation tasks involved in the preparation of labeled dataset.

In this study, the research team approached the incident detection problem using semi-supervised techniques. Maximum likelihood estimation-based contrastive pessimistic likelihood estimation (CPLE) was used for trajectory classification and identification of incident trajectories. Vehicle detection was performed using state-of-the-art deep learning-based YOLOv3, and SORT was used for tracking. Results showed that CPLE-based trajectory classification outperforms the

traditional semi-supervised techniques (self learning and label spreading) and its supervised counterpart by a significant margin.

In future work, this framework can be extended to enable operation on a network of cameras to improve detection rates and reduce false alert rates in incident detection. While the proposed framework can work at about 50 fps, extensive research needs to be done if this framework is to be implemented at a statewide level involving hundreds or thousands of cameras.

Additionally, the performance of integrated detection-and-tracking algorithms can also be explored to enable better trajectory estimation and thereby improve the accuracy of the algorithm.

REFERENCES

Adu-Gyamfi, Y., S. Asare, A. Sharma, and T. Titus. 2017. Automated Vehicle Recognition with Deep Convolutional Neural Networks. *Transportation Research Record: Journal of the Transportation Research Board*, Vol. 2645, No. 1, pp. 113–122. doi:10.3141/2645-13.

Anbaroglu, B., B. Heydecker, and T. Cheng. 2014. Spatio-Temporal Clustering for Non-Recurrent Traffic Congestion Detection on Urban Road Networks. *Transportation Research Part C: Emerging Technologies*, Vol. 48, pp. 47–65. doi:10.1016/j.trc.2014.08.002.

Asmaa, O., K. Mokhtar, and O. Abdelaziz. 2013. Road Traffic Density Estimation Using Microscopic and Macroscopic Parameters. *Image and Vision Computing*, Vol. 31, No. 11, pp. 887–894. doi:10.1016/j.imavis.2013.09.006.

Bachmann, C., B. Abdulhai, M. J. Roorda, and B. Moshiri. 2013. A Comparative Assessment of Multi-Sensor Data Fusion Techniques for Freeway Traffic Speed Estimation Using Microsimulation Modeling. *Transportation Research Part C: Emerging Technologies*, Vol. 26, pp. 33–48. doi:10.1016/j.trc.2012.07.003.

Balakrishnama, S. and A. Ganapathiraju. 1998. *Linear Discriminant Analysis—A Brief Tutorial*. Institute for Signal and Information Processing, Mississippi State University, MS.

Balcilar, M. and A. Coşkun Sönmez. 2008. Extracting Vehicle Density from Background Estimation Using Kalman Filter. In *2008 23rd International Symposium on Computer and Information Sciences, ISCIS 2008*. doi:10.1109/ISCIS.2008.4717950.

Bauza, R., J. Gozalvez, and J. Sanchez-Soriano. 2010. Road Traffic Congestion Detection through Cooperative Vehicle-to-Vehicle Communications. In *Proceedings - Conference on Local Computer Networks, LCN*, 606--612. doi:10.1109/LCN.2010.5735780.

Ben-David, S., T. Lu, and D. Pál. 2008. Does Unlabeled Data Probably Help? {W}orst-Case Analysis of the Sample Complexity of Semi-Supervised Learning. In *21st Annual Conference on Learning Theory COLT*, pp. 33–44.

Bengio, Y., O. Delalleau, and N. Le Roux. 2006. Label Propagation and Quadratic Criterion. In *Semi-Supervised Learning*. The MIT Press, Cambridge, MA.

Bewley, A., Z. Ge, L. Ott, F. Ramos, and B. Upcroft. n.d. Simple Online and Realtime Tracking. In *Proceedings of the International Conference on Image Processing, ICIP*, pp. 3464–3468. doi:10.1109/ICIP.2016.7533003.

Bolkensteyn, D. 2016. Vatic.js. *GitHub Repository*. GitHub.

Chakraborty, P., Y. O. Adu-Gyamfi, S. Poddar, V. Ahsani, A. Sharma, and S. Sarkar. 2018a. Traffic Congestion Detection from Camera Images Using Deep Convolution Neural Networks. *Transportation Research Record: Journal of the Transportation Research Board*, Vol. 2672, No. 45, pp. 222–231. doi:10.1177/0361198118777631.

Chakraborty, P., A. Sharma, and C. Hegde. 2018b. Freeway Traffic Incident Detection from Cameras: A Semi-Supervised Learning Approach. In *IEEE Conference on Intelligent Transportation Systems, Proceedings, ITSC*. doi:10.1109/ITSC.2018.8569426.

Choi, K. and Y. Chung. 2010. A Data Fusion Algorithm for Estimating Link Travel Time. *Journal of Intelligent Transportation Systems*, Vol. 7, No. 3–4, pp. 235–260. doi:10.1080/714040818.

Chong, Y. S. and Y. H. Tay. 2017. Abnormal Event Detection in Videos Using Spatiotemporal Autoencoder. In *International Symposium on Neural Networks*, pp. 189–196.

Dai, J., H. Qi, Y. Xiong, Y. Li, G. Zhang, H. Hu, and Y. Wei. 2017. Deformable Convolutional Networks. In *Proceedings of the IEEE International Conference on Computer Vision*. doi:10.1109/ICCV.2017.89.

Darwish, T. and K. Abu Bakar. 2015. Traffic Density Estimation in Vehicular Ad Hoc Networks: A Review. *Ad Hoc Networks*, Vol. 24, Part A, pp. 337–351. doi:10.1016/j.adhoc.2014.09.007.

Dowling, R., A. Skabardonis, M. Carroll, and Z. Wang. 2004. Methodology for Measuring Recurrent and Nonrecurrent Traffic Congestion. *Transportation Research Record: Journal of the Transportation Research Board*, No. 1867, pp. 60–68. doi:10.3141/1867-08.

Feng, Y., J. Hourdos, and G. A. Davis. 2014. Probe Vehicle Based Real-Time Traffic Monitoring on Urban Roadways. *Transportation Research Part C: Emerging Technologies*, Vol. 40, pp. 160–78. doi:10.1016/j.trc.2014.01.010.

Girshick, R., J. Donahue, T. Darrell, and J. Malik. 2014. Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 580–587. doi:10.1109/CVPR.2014.81.

Gonçalves, W. N., B. B. Machado, and O. M. Bruno. 2011. Spatiotemporal Gabor Filters: A New Method for Dynamic Texture Recognition. In *Workshop on Computer Vision*. http://arxiv.org/abs/1201.3612.

Han, J., D. Zhang, G. Cheng, N. Liu, and D. Xu. 2018. Advanced Deep-Learning Techniques for Salient and Category-Specific Object Detection: A Survey. *IEEE Signal Processing Magazine*, Vol. 35, No. 1, pp. 84–100. doi:10.1109/MSP.2017.2749125.

Hasan, M., J. Choi, J. Neumann, A. K. Roy-Chowdhury, and L. S. Davis. 2016. Learning Temporal Regularity in Video Sequences. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 733–742.

He, K., G. Gkioxari, P. Dollar, and R. Girshick. 2017. Mask RCNN. In *Proceedings of the IEEE International Conference on Computer Vision*, pp. 2961–2969. doi:10.1109/ICCV.2017.322.

He, K., X. Zhang, S. Ren, and J. Sun. 2016. Deep Residual Learning for Image Recognition. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 770–778. doi:10.1109/CVPR.2016.90.

Hui, Z., X. Yaohua, M. Lu, and F. Jiansheng. 2014. Vision-Based Real-Time Traffic Accident Detection. In *Proceeding of the 11th World Congress on Intelligent Control and Automation*, pp. 1035–38. IEEE. doi:10.1109/WCICA.2014.7052859.

Kim, C., F. Li, A. Ciptadi, and J. M. Rehg. 2015. Multiple Hypothesis Tracking Revisited. In *Proceedings of the IEEE International Conference on Computer Vision*, pp. 4696–4704. doi:10.1109/ICCV.2015.533.

Kotzenmacher, J., E. D. Minge, and B. Hao. 2004. Evaluation of Portable Non-Intrusive Traffic Detection System. *IMSA Journal*, pp. 34–42. http://www.imsasafety.org/journal/mj05/mj0504.pdf.

Lempitsky, V. and A. Zisserman. 2010. Learning to Count Objects in Images. *Advances in Neural Information Processing Systems*, pp. 1324–1332.

Lin, T. Y., M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick. 2014. Microsoft COCO: Common Objects in Context. In *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, LNCS Vol. 8693, pp. 740–755. doi:10.1007/978-3-319-10602-1_48.

Liu, W., D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg. 2016. SSD: Single Shot Multibox Detector. In *European Conference on Computer Vision*, pp. 21–37.

Loog, M. 2016. Contrastive Pessimistic Likelihood Estimation for Semi-Supervised Classification. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 38, No. 3, pp. 462–475. doi:10.1109/TPAMI.2015.2452921.

Lou, J., Q. Liu, T. Tan, and W. Hu. 2002. Semantic Interpretation of Object Activities in a Surveillance System. In *Proceedings of the International Conference on Pattern Recognition*, pp. 777–780. doi:10.1109/ICPR.2002.1048115.

Naphade, M., M. C. Chang, A. Sharma, D. C. Anastasiu, V. Jagarlamudi, P. Chakraborty, T. Huang, et al. 2018. The 2018 NVIDIA AI City Challenge. In *IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pp. 53–60. doi:10.1109/CVPRW.2018.00015.

Noland, R. B and J. W. Polak. 2002. Travel Time Variability: A Review of Theoretical and Empirical Issues. *Transport Reviews*, Vol. 22, No. 1, pp. 39–54. doi:10.1080/01441640010022456.

Oñoro-Rubio, D. and R. J. López-Sastre. 2016. Towards Perspective-Free Object Counting with Deep Learning. In *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, LNCS Vol. 9911, pp. 615–629. doi:10.1007/978-3-319-46478-7_38.

Owens, N., A. Armstrong, P. Sullivan, C. Mitchell, D. Newton, R. Brewster, and T. Trego. 2010. Traffic Incident Management Handbook. *Public Roads*, Vol. 172, pg. 116.

Ozbay, K. and P. Kachroo. 1999. *Incident Management in Intelligent Transportation Systems*. Artech House Publishers, Norwood, MA.

Ozkurt, C. and F. Camci. 2009. Automatic Traffic Density Estimation and Vehicle Classification for Traffic Surveillance Systems Using Neural Networks. *Mathematical and Computational Applications*, Vol. 14, No. 3, pp. 187–196.

Persaud, B. N. and F. L. Hall. 1989. Catastrophe Theory and Patterns in 30-Second Freeway Traffic Data- Implications for Incident Detection. *Transportation Research Part A: General*, Vol. 23, No. 2, pp. 103–113. doi:10.1016/0191-2607(89)90071-X.

Piciarelli, C., C. Micheloni, and G. L. Foresti. 2008. Trajectory-Based Anomalous Event Detection. *IEEE Transactions on Circuits and Systems for Video Technology*, Vol. 18, No. 11, pp. 1544–1554. doi:10.1109/TCSVT.2008.2005599.

Ravinder, K., S. Velmurugan, and S. Gangopadhyay. 2008. Efficacy of Video Incident Detection System for Advanced Traffic Management Under Non-Adherence of Lane Discipline. In *World Congress on Intelligent Transport Systems and ITS America's Annual Meeting*.

Redmon, J., S. Divvala, R. Girshick, and A. Farhadi. 2016. You Only Look Once: Unified, Real-Time Object Detection. *Proceedings of the IEEE International Conference on Computer Vision*. doi:10.1109/CVPR.2016.91.

Redmon, J. and A. Farhadi. 2017. YOLO9000:Better , Faster , Stronger. In *Proceedings of the IEEE International Conference on Computer Vision*. doi:10.1109/CVPR.2017.690.

———. 2018. *YOLOv3: An Incremental Improvement*. University of Washington, Seattle, WA. https://arxiv.org/pdf/1804.02767.pdf. Also uploaded to Cornell University's page at https://arxiv.org/abs/1804.02767.

Ren, J., Y, Chen, L. Xin, J. Shi, B. Li, and Y, Liu. 2016. Detecting and Positioning of Traffic Incidents via Video-Based Analysis of Traffic States in a Road Segment. *IET Intelligent Transport Systems*, Vol. 10, No. 6. pp. 428–437. doi:10.1049/iet-its.2015.0022.

Ren, S., K. He, R. Girshick, and J. Sun. 2017. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence,* Vol. 39, No. 6, pp. 1137–1149. doi:10.1109/TPAMI.2016.2577031.

Rezatofighi, S. H., A. Milan, Z. Zhang, Q. Shi, A. Dick, and I. Reid. 2015. Joint Probabilistic Data Association Revisited. In *Proceedings of the IEEE International Conference on Computer Vision*, pp. 3047–3055. doi:10.1109/ICCV.2015.349.

Rublee, E., V. Rabaud, K. Konolige, and G. Bradski. 2011. ORB: An Efficient Alternative to SIFT or SURF. In *Proceedings of the IEEE International Conference on Computer Vision*, pp. 2564–71. doi:10.1109/ICCV.2011.6126544.

Sadeky, S., A. Al-Hamadiy, B. Michaelisy, and U. Sayed. 2010. Real-Time Automatic Traffic Accident Recognition Using HFG. In *International Conference on Pattern Recognition (ICPR)*, pp. 3348–3351. doi:10.1109/ICPR.2010.817.

Schrank, D. L. and T. J. Lomax. 2007. *The 2007 Urban Mobility Report*. Texas Transportation Institute, The Texas A&M University System, College Station, TX.

Sussman, J. M. 2005. *Perspectives on Intelligent Transportation Systems (ITS)*. Springer.

Texas Transportation Institute with Cambridge Systematics, Inc. 2005. *Travel Time Reliability: Making It There on Time, All the Time.* https://transops.s3.amazonaws.com/uploaded_files/FHWA-HOP-06-070-Travel-Time-Reliability-Brochure.pdf.

Van Lint, J. W. C. and S. P. Hoogendoorn. 2010. A Robust and Efficient Method for Fusing Heterogeneous Data from Traffic Sensors on Freeways. *Computer-Aided Civil and Infrastructure Engineering,* Vol. 25, No. 8, pp. 596–612. doi:10.1111/j.1467-8667.2009.00617.x.

Vondrick, C., D. Patterson, and D. Ramanan. 2013. Efficiently Scaling up Crowdsourced Video Annotation. *International Journal of Computer Vision,* Vol. 101, No. 1, pp. 184–204.

Wojke, N., A. Bewley, and D. Paulus. 2017. Simple Online and Realtime Tracking with a Deep Association Metric. In *Proceedings of the IEEE International Conference on Image Processing (ICIP)*. doi:10.1109/ICIP.2017.8296962.

Yang, K., X. Wang, and R. Yu. 2018. A Bayesian Dynamic Updating Approach for Urban Expressway Real-Time Crash Risk Evaluation. *Transportation Research Part C: Emerging Technologies,* Vol. 96, pp. 192–207. doi:10.1016/j.trc.2018.09.020.

Yuan, Y., D. Wang, and Q. Wang. 2017. Anomaly Detection in Traffic Scenes via Spatial-Aware Motion Reconstruction. *IEEE Transactions on Intelligent Transportation Systems,* Vol. 18, No. 5, pp. 1198–1209. doi:10.1109/TITS.2016.2601655.

Zhang, C., H. Li, X. Wang, and X. Yang. 2015. Cross-Scene Crowd Counting via Deep Convolutional Neural Networks. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 833–841. doi:10.1109/CVPR.2015.7298684.

Zhao, B., L. Fei-Fei, and E. P. Xing. 2011. Online Detection of Unusual Events in Videos via Dynamic Sparse Coding. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 3313–3320. doi:10.1109/CVPR.2011.5995524.

Zhong, M. and G. Liu. 2007. Establishing and Managing Jurisdiction-Wide Traffic Monitoring Systems: North American Experiences. *Journal of Transportation Systems Engineering and Information Technology*, Vol. 7, No. 6, pp. 25–38. doi:10.1016/S1570-6672(08)60002-1.

Zhou, D., O. Bousquet, T. N. Lal, J. Weston, and B. Schölkopf. 2004. Learning with Local and Global Consistency. In *Advances in Neural Information Processing Systems*, pp. 321–28.

Zhu, X. and Z. Ghahramani. 2002. *Learning from Labeled and Unlabeled Data with Label Propagation*. School of Computer Science, Carnegie Mellon University, Pittsburg, PA.

Zhu, X. and A. B. Goldberg. 2009. Introduction to Semi-Supervised Learning. *Synthesis Lectures on Artificial Intelligence and Machine Learning*, Vol. 3, No. 1, pp. 1–130.

THE INSTITUTE FOR TRANSPORTATION IS THE FOCAL POINT FOR TRANSPORTATION AT IOWA STATE UNIVERSITY.

InTrans centers and programs perform transportation research and provide technology transfer services for government agencies and private companies;

InTrans manages its own education program for transportation students and provides K-12 resources; and

InTrans conducts local, regional, and national transportation services and continuing education programs.

IOWA STATE UNIVERSITY

Visit www.InTrans.iastate.edu for color pdfs of this and other research reports.