Freeway Traffic Incident Detection from Cameras: A Semi-Supervised Learning Approach

Pranamesh Chakraborty[1], Anuj Sharma[2], and Chinmay Hegde[3]

Abstract— **Early detection of incidents is a key step to reduce incident related congestion. State Department of Transportation (DoTs) usually install a large number of Close Circuit Television (CCTV) cameras in freeways for traffic surveillance. In this study, we used semi-supervised techniques to detect traffic incident trajectories from the cameras. Vehicle trajectories are identified from the cameras using state-of-the-art deep learning based You Look Only Once (YOLOv3) classifier and Simple Online Realtime Tracking (SORT) is used for vehicle tracking. Our proposed approach for trajectory classification is based on semi-supervised parameter estimation using maximum-likelihood (ML) estimation. The ML based Contrastive Pessimistic Likelihood Estimation (CPLE) attempts to identify incident trajectories from the normal trajectories. We compared the performance of CPLE algorithm to traditional semi-supervised techniques Self Learning and Label Spreading, and also to the classification based on the corresponding supervised algorithm. Results show that approximately 14% improvement in trajectory classification can be achieved using the proposed approach.**

Index Terms— **traffic incident detection, surveillance, semi-supervised learning, maximum likelihood estimation**

I. Introduction

Quicker traffic incident detection in freeways is critical for providing rapid incident response. Studies have shown that every seven minutes of delay in incident verification leads to an additional mile of queue build-up [1], thereby increasing the likelihood of secondary incidents. Improved procedures for incident management resulted in reduction of $3.06 million and 143.3 million hours of incident-related congestion [2]. Hence, significant efforts have been devoted towards development of accurate and fast automatic incident detection (AID) algorithms. Traditional AID algorithms rely on radar-based sensor data [3], loop detector data [4], probe vehicle data [5], or fusing multiple multiple streams [6], [7] to detect traffic incidents from data streams. State Departments of Transportation (DoTs) also install a large number of Closed Circuit Television (CCTV) cameras for surveillance tasks along freeways. Traffic incident detection approaches from CCTV cameras can be broadly classified into two categories: (a) Explicit Event Recognition and (b) Anomaly Detection.

In explicit event recognition, the explicit knowledge of the events to be identified are used for incident detection. This requires *a priori* knowledge of all the recognizable events, which the associated AID systems use as predefined templates to parse the incoming data for incident detection. For example, Ravinder et al. applied video image processing for traffic management where there is non-adherence to lane discipline [8]. Sadek et al. used logistic regression over Histogram of Flow Gradients (HFG) to determine the probability of occurrence of accident in a video sequence [9]. Zu et al. used Gaussian Mixture Model (GMM) to detect traffic vehicles and tracked using Mean Shift Algorithm. Then traffic incident alarms were triggered when the velocity or acceleration of the detected vehicles exceed a pre-determined threshold [10]. Ren et al. used video based detection for analyzing the traffic state distribution characteristics in a cluster of cells dividing the lanes of a road segment [11]. More recently, Chakraborty et al. used deep convolutional neural networks for detecting traffic congestion in camera images [12]. Maaloul et al. used Farneback Optical Flow for motion detection in a video sequence, together with a heuristic threshold-selection approach for accident detection [13].

The other popular approach to incident detection is based on 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. used dynamic clustering techniques to cluster the normal trajectories and detect the abnormal ones. [14]. Picarelli used one-class Support Vector Machine (SVM) for detecting anomalous trajectories [15]. Recently, Yuan et al. performed anomaly detection in traffic scenes using spatially-aware motion reconstruction [16]. Such unsupervised modeling of the video sequences also traditionally involved a combination of sparse coding and bag-of-words (BOG) [17]. However, recent developments in deep learning techniques have resulted in new methods learning normal video patterns and thereby detecting anomalies based on reconstruction error. Hasan et al. used a fully convolutional feed-forward autoencoder to learn the spatio-temporal local features and thereby learn the temporal regularity in the video sequences [18]. Chong and Tay also used a combination

*Our research results are based upon work jointly supported by the National Science Foundation Partnerships for Innovation: Building Innovation Capacity (PFI: BIC) program under Grant No. 1632116, National Science Foundation under Grants No. CNS-1464279 and CCF-1566281, and Iowa DOT Office of Traffic Operations Support Grant. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

[1]Pranamesh Chakraborty is a PhD student in the Department of Civil, Construction and Environmental Engineering, Iowa State University, Iowa, USA pranames@iastate.edu

[2]Anuj Sharma is an Associate Professor in the the Department of Civil, Construction and Environmental Engineering, Iowa State University, Iowa, USA anujs@iastate.edu

[3]Chinmay Hegde is an Assistant Professor in the Department of Electrical and Computer Engineering, Iowa State University, Iowa, USA chinmay@iastate.edu

of spatial feature extractor and temporal sequencer based on Convolutional Long Short Term Memory (ConvLSTM) network for anomaly detection in videos [19].

Thus, broadly, the above two categories can 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 its application is the scarcity of enough supervised data samples and 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 work, we establish a new learning framework for traffic incident detection using recent advances in semi-supervised learning [20]. Via this framework, we achieve the "best-of-both-worlds"; we manually annotate only a small sample of normal vehicle tracks and tracks of vehicles involved in an incident, and then we used all other (unlabeled) vehicle tracks to improve the classification performance. Our experimental results on traffic data provided to us by the Iowa DoT demonstrate that our framework achieves superior performance compared to supervised learning techniques with a comparable amount of labeled examples.

The paper is structured as follows. Section II contains the details of the methodology adopted for vehicle detection, tracking, and the semi-supervised techniques for vehicle trajectory classification. Section III gives the details of the data used in this study followed by the detailed results in Section IV. The final section provides a summary of the paper and outlines avenues for future work.

II. Methodology

Traffic incident detection from videos using trajectory information comprises of 3 basic tasks: (a) vehicle detection (b) vehicle tracking and trajectory formation, and (c) trajectory classification. Each task is described next, with our primary focus geared towards trajectory classification using semi-supervised techniques.

A. Vehicle Detection

In recent years, evolution of convolutional neural networks (CNN) have resulted in significant improvement in object detection and classification performance. In this study, we chose YOLOv3 [21] for vehicle detection primarily because of its fast performance with reasonable accuracy which makes it suitable for real-time performance. Specifically, we used the YOLOv3-416 model trained on the Microsoft Common Objects in Context (COCO) dataset [22]. We used the classes: 'car', 'motorbike', 'bus', and 'truck' out of the 80 classes in the COCO dataset for our vehicle detection module.

B. Vehicle Tracking and Trajectory Formation

Recent improvements in object detection performances have led to tracking-by-detection as the leading paradigm for multi-object tracking (MOT). In MOT, multiple objects are detected in each frame and the aim is to associate the detections across frames in a video sequence. In this study we used the Simple Online and Realtime Tracking (SORT) algorithm for vehicle tracking. This is an online multi-object tracking algorithm which uses the Kalman Filter and the Hungarian algorithm for the data association problem. We chose this tracker because of its reasonable performance in online, realtime settings. SORT tracker updates at 260 Hz, making it suitable for realtime implementation.

Our object tracker module outputs a sequence of bounding box coordinates, X-center (X^c), 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 2-dimensional points, denoted as $TR_i = (p_1, p_2, p_3 \dots p_j \dots p_{len_i})$. Here, each p_j is a 2-dimensional point representing the bounding box coordinates. The length len_i of a trajectory can be different for different trajectories. Note that, in our study, we consider only the bounding box center coordinates, but other features such as bounding box appearance descriptors can also be included.

C. Semi Supervised Trajectory Classification

The aim of semi-supervised learning is to exploit the 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 the supervised counterparts. On the contrary, empirical studies have observed severely deteriorated performances [23]. Recently, Loog demonstrated how Maximum Likelihood (ML) can be used to improve classification performance in semi-supervised setting [20].

In this paper, we address the problem of trajectory classification in semi-supervised settings using the Contrastive Pessimistic Likelihood Estimation (CPLE) based on ML estimation [20]. We present experimental results proving the validity of our approach and compare the results with the traditional semi-supervised classification techniques. We discuss next the details of the CPLE method for semi-supervised classification followed by a brief description of the traditional algorithms that we chose for comparison.

1) 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 is modeled as behave adversarially so that any semi-supervised learning mechanism least benefits from it. This makes it 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:

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

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 $x_i \in \mathbb{R}^d d$-dimensional feature vectors, and $y_i \in C = \{1, ..., K\}$ are their corresponding labels.

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

$$\widehat{\theta}_{\text{sup}} = \underset{\theta}{\operatorname{argmax}}\, L\left(\theta | X\right). \tag{2}$$

In our study, we chose Linear Discriminant Analysis (LDA) as our classifier, similar to the approach of Loog [20]. Here, the log-likelihood objective function is given by

$$\begin{aligned} L_{LDA}\left(\theta|X\right) &= \sum_{i=1}^{N} \log p\left(x_i, y_i | \pi_1, ..., \pi_k, \mu_1, ..., \mu_k, \Sigma\right) \\ &= \sum_{k=1}^{K}\sum_{j=1}^{N_k} \log p\left(x_{kj}, k | \pi_k, \mu_k, \Sigma\right), \end{aligned} \tag{3}$$

where $\theta = (\pi_1, ..., \pi_k, \mu_1, ..., \mu_k, \Sigma)$, 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 data set by

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

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

$$\widehat{\theta}_{\text{opt}} = \underset{\theta}{\operatorname{argmax}}\, L\left(\theta | X_V\right) \tag{4}$$

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

$$L\left(\widehat{\theta}_{\text{sup}} | X_V\right) \leq L\left(\widehat{\theta}_{\text{opt}} | X_V\right) \tag{5}$$

In semi-supervised setting, V is unobserved, but we have X (labeled data) and U (unlabeled data). We have more information compared to supervised setting but less than the fully labeled case. Thus,

$$L\left(\widehat{\theta}_{\text{sup}} | X_V\right) \leq L\left(\widehat{\theta}_{\text{semi}} | X_V\right) \leq L\left(\widehat{\theta}_{\text{opt}} | X_V\right) \tag{6}$$

Now, we take the supervised estimate into account explicitly in order to construct a semi-supervised classifier than can improve upon its supervised counterpart.

Before doing so, we define q_{ki} 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 $\mathbb{R}^K$:

$$q_{\cdot i} \in \Delta_{K-1} = \left\{ \left(\rho_1, \ldots, \rho_K\right)^T \in \mathbb{R}^K | \sum_{i=1}^{K} \rho_i = 1, \rho_i \geq 0 \right\} \tag{7}$$

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

$$L\left(\theta | X, U, q\right) = L\left(\theta | X\right) + \sum_{i=1}^{M}\sum_{k=1}^{K} q_{ki} \log p\left(u_i, k | \theta\right) \tag{8}$$

where the variable q in 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

$$CL\left(\theta, \widehat{\theta}_{\text{sup}} | X, U, q\right) = L\left(\theta | X, U, q\right) - L\left(\widehat{\theta}_{\text{sup}} | X, U, q\right) \tag{9}$$

This enables us to check the extent of improvement of semi-supervised estimates in terms of log-likelihood, defined as *contrast*. Since q is unknown, Equation 9 cannot be used directly in optimization. Hence, we choose the most pessimistic solution where we assume that the true (soft) labels achieve the worst-case among all semi-supervised solutions and consider the q which minimizes the likelihood gain. Thus, our objective function can be written as:

$$CPL\left(\theta, \widehat{\theta}_{\text{sup}} | X, U\right) = \min_{q \in \Delta_{K-1}^{M}} CL\left(\theta, \widehat{\theta}_{\text{sup}} | X, U, q\right) \tag{10}$$

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:

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

$$\widehat{\pi}_k = \frac{N_k + \sum_{i=1}^{M} q_{ki}}{N + M} \tag{11}$$

$$\widehat{\mu}_k = \frac{\sum_{j=1}^{N_k} x_{kj} + \sum_{i=1}^{M} q_{ki} u_i}{N_k + \sum_{i=1}^{M} q_{ki}} \tag{12}$$

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

2) 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 to restricted to 3,000.

2) Baseline algorithms: We compared the performance of our above CPLE-based framework for trajectory classification with respect to two baseline semi-supervised methods: Self Learning [24] and Label Spreading [25]. Self Learning combines information from unlabeled data with the labeled data to iteratively identify the label of unlabeled data. The labeled training set is enlarged on each iteration until the entire dataset is labeled. LDA [26] is used as the base model in Self Learning in this study. Label Spreading [25], a modification of the traditional Label Propagation algorithm [27] 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 [28] for further details. Besides these two baseline, we also compared the results of our algorithm to its supervised counterpart obtained from the LDA classifier trained on the labeled data.

3) Feature Vector Generation: The trajectories obtained from the vehicle tracker module are of variable length (see Section II-B). However, the semi-supervised techniques described above requires fixed-dimensional feature vectors. Hence, we first used trajectory subsampling to convert these variable length trajectories into fixed-length trajectories, similar to [15]. Each trajectory is subsampled to form a list of 2-D coordinates. We heuristically chose 75 as the fixed length of each of these lists. since the typical length of each trajectory is between 70 to 80. Thus, each trajectory can now be defined as $TR_i = p_1 p_2 p_3 ... p_j ... p_{75}$, where p_j is the 2D vector representing $[X^c_j, Y^c_j]$. We normalized the feature vector to zero mean and performed Principal Component Analysis for dimension reduction. We found that 95% of variance (explained by top 3 principal components for X^c and Y^c each) is sufficient, similar to [20]. Finally, the top 3 principal components for X^c and Y^c are concatenated to form a 6-D vector representing the trajectory information of each vehicle id. This 6-D feature vector is used for trajectory classification.

III. Data Description

The primary source of data in this study are the traffic incident videos obtained from the CCTV cameras installed by the Iowa Department of Transportation (DoT) along the freeways of Iowa. Our dataset consists of 151 traffic incident videos recorded from these cameras during the period of 2016-2017. Each video is of two-minutes duration recorded at 30 frames per second and clearly captures the onset of the traffic incident. The resolution of videos varies from 800×480 pixels to 1920×1080 pixels depending on the camera resolution. The traffic incidents are caused due to car crashes or stalled vehicles. Out of the 151 incident videos, we manually annotated 11 videos with the bounding boxes of the vehicles involved in the incident. A javascript based video annotation tool [29] based on VATIC [30] is used for annotating the vehicles. This resulted in a total of 15 unique trajectories of vehicles involved in incidents. These trajectories are then matched with the vehicle trajectories obtained from object detection and tracking modules used in this study (YOLOv3 for vehicle detection and SORT for vehicle tracking). For each frame, each manually annotated bounding box is matched with the detected bounding box with maximum overlap, setting a minimum threshold of 0.5 Intersection over Union (IoU).Each manually annotated incident trajectory was successfully matched with unique trajectory obtained from the tracking algorithm. These trajectories are henceforth referred to as incident trajectories. The remaining trajectories in those 11 incident videos are classified as normal trajectories. We randomly selected 15 such normal trajectories into our labeled dataset. Thus, our labeled dataset consists of 15 normal trajectories and 15 incident trajectories.

We randomly selected 90 incident videos from the 151 incident videos for our unlabeled dataset preparation. The 11685 trajectories obtained by the object detection and tracking algorithm from those 90 videos are included in the unlabeled dataset. The remaining 50 incident videos are equally divided into validation and test data sets, with 25 incident videos in each set. We also randomly selected 50 baseline videos without any incidents and split them equally into validation and test set. Thus, our validation dataset and test dataset consist of 50 videos each, 25 of them being incident videos and remaining 25 being normal baselines videos. Our validation and test datasets consist of 6333 and 5375 trajectories respectively.

IV. Results

We used the state-of-the-art object detection algorithm YOLOv3 [21] for vehicle detection and SORT [31] for vehicle tracking. The object detection and tracking runs at around 45 frames per second (fps) on an NVIDIA GTX 1080 GPU, making it suitable for real-time performance. Figure 1 shows a sample image of vehicle trajectories.

Fig. 1. Vehicle detection and tracking sample

Our labeled trajectory dataset consists of 15 incident trajectories and 15 normal trajectories. To find out the sensitivity of the algorithm on the number of labeled examples, we ran each algorithm for label sample sizes varying from 5-15 trajectories for each class (normal and incident). The efficacy of the proposed model (CPLE) along with the comparison models (Label Spreading, Self Learning and Supervised Learning) are validated using the validation dataset and the final accuracy is reported for the test dataset. We label a video as an incident video if at least 1 trajectory in the video is classified as incident trajectory by the algorithm. The accuracy of the algorithm (ACC) is given by the accuracy of correctly classifying incident videos (TPR) and baseline videos (TNR), as shown in Equations 15, 13, and 14. TP and TN refer to the number of correctly identified incident and baseline videos while P and N refer to total number of incident and baseline videos (25 each).

$$TPR = \frac{TP}{P} \tag{13}$$

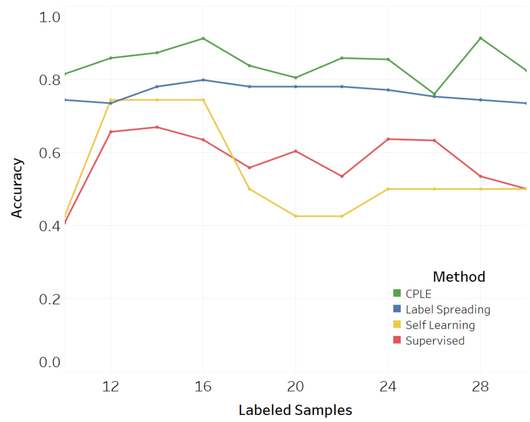

Fig. 2. Accuracy of the algorithms on number of labeled samples

TABLE I

ACCURACY OF ALGORITHMS ON TEST DATA

Method	TPR	TNR	ACC
CPLE	0.83	0.92	0.88
Label Spreading	0.6	0.94	0.77
Self Learning	0.53	0.88	0.71
Supervised	0.28	0.96	0.62

$$TNR = \frac{TN}{N} \tag{14}$$

$$ACC = \frac{TP + TN}{P + N} \tag{15}$$

Figure 2 shows the accuracy of each algorithm on the validation dataset for different number of labeled samples. The experiments are repeated 20 times and the average accuracy is reported. We clearly see that CPLE performs superior compared to the other semi-supervised approaches and its supervised counterpart. On an average, 14% improvement is obtained on using CPLE compared to the second best algorithm (Label Spreading). The best model obtained from each algorithm is selected and applied on the test dataset. Table I shows the accuracy of each algorithm on the test dataset. It shows that while CPLE successfully identifies a large majority of the incident videos (21 out 25 incident videos), other algorithms fail to do so and perform poorly in TPR. However, since majority of trajectories are normal trajectories, all algorithms perform well in detecting the baseline videos correctly. This shows that the CPLE algorithm successfully extracts information regarding both incident and normal trajectories from the unlabeled dataset and hence achieves better performance.

Figure 3 shows a sample of incident and normal trajectories labeled by the CPLE algorithm for 3 incident videos (ID 1, 2, and 3). The X and Y coordinates of the bounding box center of each vehicle across the video frames is shown in the figure. It successfully detects the incident trajectories in Video ID 1 and 2, but fails to detect any incident trajectory in ID 3, primarily due to missing object detection caused by poor video quality. A sample image of a stalled vehicle detected across 3 frames is shown in Figure 4.

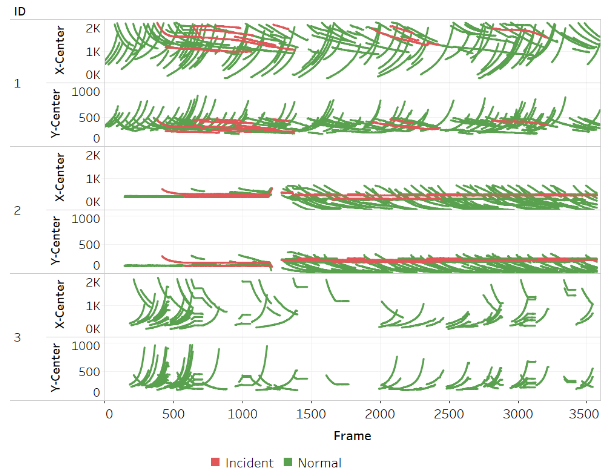

Fig. 3. Incident and normal trajectories labeled by CPLE algorithm for 3 incident videos

V. CONCLUSIONS

State Department of Transportation (DOTs) usually install a large number of CCTV cameras across freeways for surveillance purpose. 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 using these these cameras. In this study, we approached the incident detection problem using semi-supervised techniques. We used Maximum Likelihood Estimation based Contrastive Pessimistic Likelihood Estimation (CPLE) for trajectory classification and identification of incident trajectories. Vehicle detection is performed using state-of-art deep learning based YOLOv3 and SORT tracker is used for tracking. Results show that CPLE based trajectory classification outperforms the traditional semi-supervised techniques (Self Learning and Label Spreading) and also its supervised counterpart by significant margin.

As future work, we intend to expand our framework to enable operation on a network of cameras to improve detection rates and reduce false alarms rates in incident detection. We also intend to explore the performance of integrated detection-and-tracking algorithms to enable better trajectory estimation.

REFERENCES

[1] T. Wells and E. Toffin, "Video-based automatic incident detection on San-Mateo bridge in the San Francisco bay area," in *12th World Congress on Intelligent Transportation Systems, San Francisco*. Citeseer, 2005.

[2] D. L. Schrank and T. J. Lomax, "The 2007 urban mobility report," Texas Transportation Institute, The Texas A&M University System, Tech. Rep., 2007.

[3] Q. Shi and M. Abdel-Aty, "Big data applications in real-time traffic operation and safety monitoring and improvement on urban expressways," *Transportation Research Part C: Emerging Technologies*, vol. 58, pp. 380–394, 2015.

[4] C. Xu, P. Liu, B. Yang, and W. Wang, "Real-time estimation of secondary crash likelihood on freeways using high-resolution loop detector data," *Transportation Research Part C: Emerging Technologies*, vol. 71, pp. 406–418, 2016.

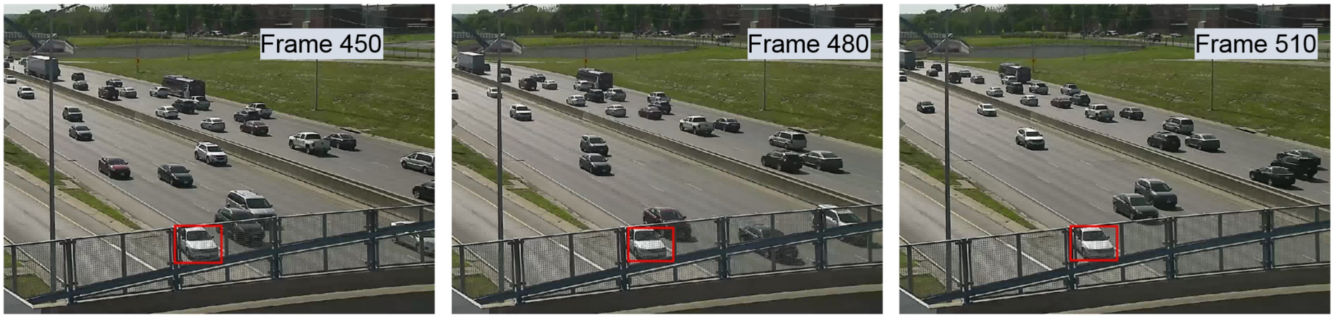

Fig. 4. Sample images of stalled vehicle detected across 3 frames taken at 1-second interval

[5] P. Chakraborty, J. R. Hess, A. Sharma, and S. Knickerbocker, "Outlier mining based traffic incident detection using big data analytics," in *Transportation Research Board 96th Annual Meeting Compendium of Papers*, 2017.

[6] H. Dia and K. Thomas, "Development and evaluation of arterial incident detection models using fusion of simulated probe vehicle and loop detector data," *Information Fusion*, vol. 12, no. 1, pp. 20–27, 2011.

[7] M. Houbraken, P. Audenaert, D. Colle, M. Pickavet, K. Scheerlinck, I. Yperman, and S. Logghe, "Real-time traffic monitoring by fusing floating car data with stationary detector data," in *2015 International Conference on Models and Technologies for Intelligent Transportation Systems (MT-ITS)*. IEEE, 2015, pp. 127–131.

[8] K. Ravinder, S. Velmurugan, and S. Gangopadhyay, "Efficacy of video incident detection system for advanced traffic management under non-adherence of lane discipline," in *Intelligent Transportation Systems Connections: Saving Time. Saving Lives*, 2008.

[9] S. Sadeky, A. Al-Hamadiy, B. Michaelisy, and U. Sayed, "Real-time automatic traffic accident recognition using hfg," in *2010 20th International Conference on Pattern Recognition (ICPR)*. IEEE, 2010, pp. 3348–3351.

[10] Z. Hui, X. Yaohua, M. Lu, and F. Jiansheng, "Vision-based real-time traffic accident detection," in *2014 11th World Congress on Intelligent Control and Automation (WCICA)*. IEEE, 2014, pp. 1035–1038.

[11] J. Ren, Y. Chen, L. Xin, J. Shi, B. Li, and Y. Liu, "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, 2016.

[12] P. Chakraborty, Y. O. Adu-Gyamfi, S. Poddar, V. Ahsani, A. Sharma, and S. Sarkar, "Traffic congestion detection from camera images using deep convolution neural networks," in *Transportation Research Board 97th Annual Meeting Compendium of Papers*, 2018.

[13] B. Maaloul, A. Taleb-Ahmed, S. Niar, N. Harb, and C. Valderrama, "Adaptive video-based algorithm for accident detection on highways," in *2017 12th IEEE International Symposium on Industrial Embedded Systems (SIES)*. IEEE, 2017, pp. 1–6.

[14] J. Lou, Q. Liu, T. Tan, and W. Hu, "Semantic interpretation of object activities in a surveillance system," in *Proceedings of 16th International Conference on Pattern Recognition*, vol. 3. IEEE, 2002, pp. 777–780.

[15] C. Piciarelli, C. Micheloni, and G. L. Foresti, "Trajectory-based anomalous event detection," *IEEE Transactions on Circuits and Systems for video Technology*, vol. 18, no. 11, pp. 1544–1554, 2008.

[16] Y. Yuan, D. Wang, and Q. Wang, "Anomaly detection in traffic scenes via spatial-aware motion reconstruction," *IEEE Transactions on Intelligent Transportation Systems*, vol. 18, no. 5, pp. 1198–1209, 2017.

[17] B. Zhao, L. Fei-Fei, and E. P. Xing, "Online detection of unusual events in videos via dynamic sparse coding," in *2011 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2011, pp. 3313–3320.

[18] M. Hasan, J. Choi, J. Neumann, A. K. Roy-Chowdhury, and L. S. Davis, "Learning temporal regularity in video sequences," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2016, pp. 733–742.

[19] Y. S. Chong and Y. H. Tay, "Abnormal event detection in videos using spatiotemporal autoencoder," in *International Symposium on Neural Networks*. Springer, 2017, pp. 189–196.

[20] M. Loog, "Contrastive pessimistic likelihood estimation for semi-supervised classification," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 38, no. 3, pp. 462–475, 2016.

[21] J. Redmon and A. Farhadi, "YOLOv3: An Incremental Improvement," *ArXiv e-prints 1804.02767*, Apr. 2018.

[22] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft coco: Common objects in context," in *European Conference on Computer Vision ECCV 2014*, D. Fleet, T. Pajdla, B. Schiele, and T. Tuytelaars, Eds. Cham: Springer International Publishing, 2014, pp. 740–755.

[23] S. Ben-David, T. Lu, and D. Pál, "Does unlabeled data probably help? Worst-case analysis of the sample complexity of semi-supervised learning." in *21st Annual Conference on Learning Theory COLT*, 2008, pp. 33–44.

[24] X. Zhu and A. B. Goldberg, "Introduction to semi-supervised learning," *Synthesis lectures on artificial intelligence and machine learning*, vol. 3, no. 1, pp. 1–130, 2009.

[25] D. Zhou, O. Bousquet, T. N. Lal, J. Weston, and B. Schölkopf, "Learning with local and global consistency," in *Advances in Neural Information Processing Systems*, 2004, pp. 321–328.

[26] S. Balakrishnama and A. Ganapathiraju, "Linear discriminant analysis-a brief tutorial," *Institute for Signal and Information Processing*, vol. 18, pp. 1–8, 1998.

[27] X. Zhu and Z. Ghahramani, "Learning from labeled and unlabeled data with label propagation," School Comput. Science, Carnegie Mellon University, Pittsburgh, PA, Tech. Rep. CMU-CALD-02-107, Tech. Rep., 2002.

[28] Y. Bengio, O. Delalleau, and N. Le Roux, *Label Propagation and Quadratic Criterion*. MIT Press, January 2006.

[29] D. Bolkensteyn, "Vatic.js," https://github.com/dbolkensteyn/vatic.js, 2016.

[30] C. Vondrick, D. Patterson, and D. Ramanan, "Efficiently scaling up crowdsourced video annotation," *International Journal of Computer Vision*, vol. 101, no. 1, pp. 184–204, 2013.

[31] A. Bewley, Z. Ge, L. Ott, F. Ramos, and B. Upcroft, "Simple online and realtime tracking," in *2016 IEEE International Conference on Image Processing (ICIP)*. IEEE, 2016, pp. 3464–3468.