Soft Computing
https://doi.org/10.1007/s00500-021-05576-w

FOCUS

Real-time image enhancement for an automatic automobile accident detection through CCTV using deep learning

Manu S. Pillai[1] · Gopal Chaudhary[1] · Manju Khari[2] · Rubén González Crespo[3]

Abstract
Almost all of the automatic accident detection (AAD) system suffers from the tradeoff between computational overhead and detection accuracy. Recent advances in detection and classification methodologies have shown phenomenal improvements in accuracy but these systems require a huge number of computational resources making them unviable for deployment requiring real-time feedback. This paper proposes a methodology to develop a reliable and computationally inexpensive real-time automatic accident detection system that can be deployed with minimum hardware requirements. Specifically, we split our AAD system into three major stages (Detection, Tracking and Classification) and propose algorithms for each stage with reduced computational need. For the detection stage, we propose Mini-YOLO, a deep learning model architecture trained using knowledge distillation that has comparable accuracy with its counterpart YOLO(You-Only-Look-Once) with reduced model size and computational overhead. Mini-YOLO achieves an average precision (AP) score of 34.2 on the MS-COCO dataset while outperforming all other detection algorithms in runtime complexity, achieving a staggering 28 frames per second on a low-end machine. For the tracking stage, we adopt SORT (Simple Online Real-time Tracking) and for classification stage, we compare multiple machine learning algorithms and show that a support vector machine with radial basis kernel performs the best with an area under the curve (AUC) score of 0.98, model size of 448 KB (kilobytes) and 12.73 ms (milliseconds) latency.

Keywords Vehicle accident detection · CCTV image processing · Knowledge distillation · YOLO · Vehicle tracking · Automatic accident detection system

1 Introduction

Nowadays, it is very essential to detect traffic accidents as early as possible for traffic safety and time-saving, worldwide. The event of traffic accidents is a major concern mainly in highly congested countries. Amidst a speedy rise in the figure of expressways and motorized vehicles in most countries (Global status report on road safety 2015), the cumulative number of collisions has risen extensively in the world. A yearly report of the National Highway Traffic Safety Administration (NHTSA) stated that about 5,000,000 traffic accidents happen in the United States (US) every year (Facts and FARS, 2013.Natl. Highw. Traffic Saf. Adm. (NHTSA), GES Annu. Rep. Natl. Highw. Traffic Saf. Adm. (NHTSA), GES Annu. Rep. 2013). In reality, it is found that traffic accidents have caused major death numbers for people within 30 and 44 in the US (Facts and FARS, 2013.Natl. Highw. Traffic Saf. Adm. (NHTSA), GES Annu. Rep. Natl. Highw. Traffic Saf. Adm. (NHTSA), GES Annu. Rep. 2013). In the Traffic Safety Facts Annual Report Tables, it is shown that death or injury occurs during road accidents have increased by up to 3% and 2%, respectively, between 2011 and 2012 (Facts and FARS, 2013.Natl. Highw. Traffic Saf. Adm. (NHTSA), GES Annu. Rep. Natl. Highw. Traffic Saf. Adm. (NHTSA), GES Annu. Rep. 2013). It was also reported by the World Health Organization that yearly 1.25 million people die in traffic

Communicated by Vicente Garcia Diaz.

✉ Rubén González Crespo
rubenagc@gmail.com

[1] Bharati Vidyapeeth's College of Engineering, New Delhi, India

[2] Netaji Subhas University of Technology, East Campus, Delhi, India

[3] Computer Science and Technology Department, Universidad Internacional de La Rioja, Logroño, Spain

Published online: 27 January 2021

accidents (Global status report on road safety 2015). It was seen if accidents are not identified correctly and swiftly, these accidents produce negative sentiments on traffic, particularly on highways, between the junctions, and in work zones, frequently ending in prolonged congestion and effusion (Azimi et al. 2019; Arvin et al. 2019; Mokhtarimousavi et al. 2019; Management and (2013) Fed. Highw. Adm. Traffic safety facts, 2012). The transport community has been frequently proceeding with the help of innovative computational techniques, machine learning, and new algorithms (Sharifi et al. 2019; Golshani et al. 2018; Parsa et al. 2019a; Nasr Esfahani et al. 2019; Razi-Ardakani et al. 2018; Ahangari et al. 2019), which have also expedited the prediction (Mansourkhaki et al. 2016, 2017), detection (Parsa et al. 2019b), and estimation of the austerity of accidents.

Recently smartphones are decked with numerous sensors that provide positional and similar data that can give data to identify accidents (Alwan et al. 2016; Fernandes et al. 2016). Examples of these sensors are accelerometers, magnetometers, and gyroscopes. The data from such smartphones have been combined with other sources of data such as NetLogo simulated data (Thomas and Vidal 2017) and airbag triggers data (Zaldivar et al. 2011) to detect accidents. However, these kinds of data are highly limited and are expensive to procure. Other potential data sources include visual data like images and videos. These image and video data use easy and cheap cameras, hence become the inexpensive source of information. Numerous researches have used these data sources along with various methods including matrix approximation, statistic heuristic method, hybrid support vector machine with extended Kalman filter, and extreme learning machine to identify accidents (Xia et al. 2015; Maaloul et al. 2017; Vishnu and Nedunchezhian 2018; Chen et al. 2016b).

Vision-based accident detection needs a huge volume of data from photos and videos. Hence, it expects a large storage capacity. Besides, the accuracy of vision-based accident detection models is affected by weather conditions and the resolution of the camera.

Some other data sources for accidental data are social media. Deep belief network (DBN) and long short-term memory (LSTM) are two deep learning models that have been adopted to detect accidents from social media. Several fusion-based techniques have been adopted to combine social media data with traffic data to achieve higher performance. Generally, social media data are assumed to be random and poor (Gu et al. 2016).

Moreover, many machine learning models have been used to detect accidents, including k-nearest neighbor, regression tree (Ozbayoglu et al. 2017), feed-forward neural network (Ozbayoglu et al. 2017), support vector machine (Dong et al. 2015), probabilistic neural network (Parsa et al. 2019), dynamic Bayesian network (Sun and Sun 2015), and deep learning (Parsa et al. 2019a). To this end, traffic data are often considering to be best suited to detect and predict the occurrence of accidents. S Naz et.al proposed a driver fatigue detection mechanism scheme where from the captured video, localization of eyes using Viola-Jones algorithm is done. Once the eyes have been localized, they are classified as open or closed using three different techniques namely mean intensity, SVM, and SIFT (Naz et al. 2019). KHM Kumar et.al proposed feature fusion to provide knowledge to the system by alternative sets of features obtained using linguistic and content-based text features (Kumar and Harish 2019).

Recent advances in detection and classification methodologies have shown phenomenal improvements in accuracy but these systems require a huge number of computational resources making them unviable for deployment requiring real-time feedback.

This paper proposes a methodology to develop a reliable and computationally inexpensive real-time automatic accident detection (AAD) system that can be deployed with minimum hardware requirements. Specifically, we split our AAD system into three major stages (Detection, Tracking and Classification) and propose algorithms for each stage with reduced computational need. For the detection stage, we propose Mini-YOLO, a deep learning model architecture trained using knowledge distillation (Hinton et al. 2015) that has comparable accuracy with its counterpart YOLO (Redmon et al. 2016) with reduced model size and computational overhead. Mini-YOLO achieves an average precision (AP) score of 34.2 on the MS-COCO (Lin et al. 2014) dataset while outperforming all other detection algorithms in runtime complexity, achieving a staggering 28 frames per second on a low-end machine. For the tracking stage, we adopt SORT (Bewley et al. 2016) and for classification stage, we compare multiple machine learning algorithms and show that a support vector machine with radial basis kernel performs the best with an AUC score of 0.98, model size of 448 KB (kilobytes) and 12.73 ms (milliseconds) latency.

1. **Key contributions of the paper**

(a) A methodology to develop a reliable and computationally inexpensive real-time automatic accident detection system is proposed.

   i. A 3-stage architecture is developed, for detection, tracking and classification of vehicles.
   ii. For each stage, computationally inexpensive models are proposed with comparable performance to its computationally heavier counterparts.

(b) Mini-YOLO, a deep learning model architecture for object detection with reduced model size and computational overhead is proposed.

The paper is divided into four sections. Section 1 deals with the introduction and related work in automatic accident detection. In Sect. 2, our methodology is presented while results of our experiments are highlighted in Sect. 3. Sections 4 and 5 conclude our work and outlines the future scope, respectively.

2 Methodology

The workflow of our system consists of 3 major stages. Figure 1 shows the top-level architecture of the proposed system. The first stage is the detection stage, where the raw input camera feed is processed to detect vehicles on the move. The detected vehicles in each frame are passed on to the second stage viz the tracking stage. The tracking stage keeps track of the detected vehicles and their status of damage throughout the continuous stream of input frames. The status of damage is an indicator variable associated with the vehicle tracks that is triggered when that vehicle is met with an accident. This indicator variable is controlled by the third stage which is a classification stage where each of the segmented vehicle images from the detection stage is classified into damaged or undamaged class. The indicator variable is triggered if the system detects a vehicle and classifies it into the damaged class given the track associated with it was previously classified into the undamaged class. These stages are further explained in the following section.

1. Stage 1- Vehicle Detection

Multiple methodologies are proposed for object detection in the literature but most of the work focuses on achieving high precision and recall trading out on computational complexity. High computational needs will increase manufacturing and maintenance costs and will negatively affect the scalability of our system. Hence, we wanted a detection algorithm that is computationally cheap and reliable to be able to deploy on the edge devices. Our use case reduces the task complexity of object detection by cutting down the number of classes to be detected to 1, i.e., Vehicle, given an input image.

Considering the necessity of high recall and computational lightness, we develop a Mini-YOLO vehicle detection algorithm that is computationally efficient and has a high recall.

Mini-YOLO-Traditional object detection systems repurpose classifiers or localizers to perform detection. These systems apply a classification model to an image in several areas and scales where high scoring areas of the image are taken into account. YOLO uses a completely different approach and analyzes the entire picture in one shot to predict object bounding boxes. Precisely, it divides the image into areas and provides bounding boxes and probabilities for each area. These bounding boxes are then weighted by the predicted probabilities. In contrast to traditional approaches, YOLO makes predictions with a single assessment making it multiple times faster than traditional approaches. Even though YOLO is significantly faster than competing methodologies for object detection, the model size and hardware requirements still raise a concern for its use in our system.

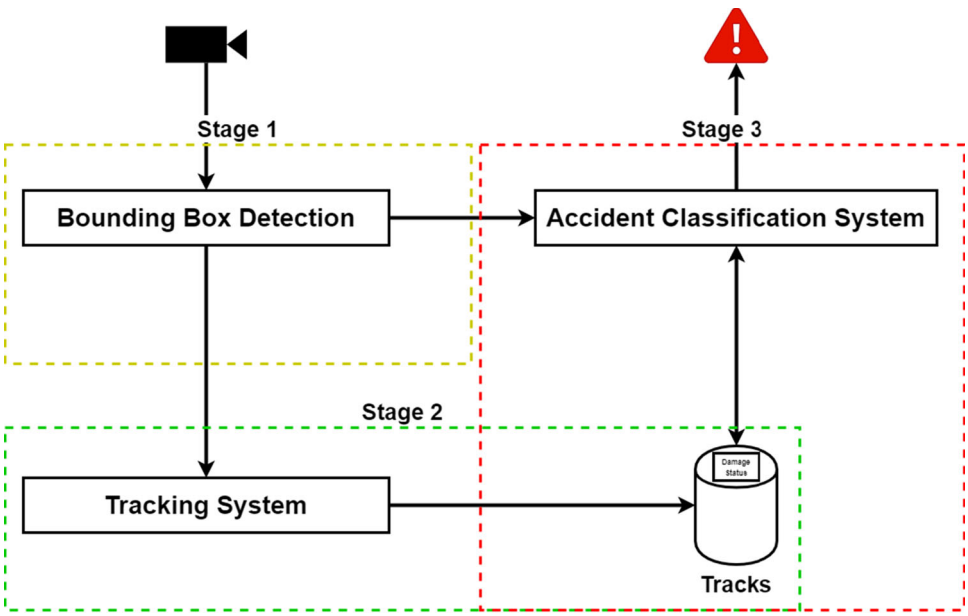

Fig. 1 Top-level architecture of the proposed system

Mini-YOLO is a distilled version of YOLO object detection algorithm and achieves comparable performance with increased computational efficiency. A distilled version of a neural network is a small version of that network architecture that is trained on the predictions of the bigger network called soft predictions. The method of creating a distilled version of a neural network is called knowledge distillation and was first proposed by Bucila et al., 2006 (Buciluǎ et al. 2006) and generalized by Hinton et al., 2015. This training setting is sometimes referred to as "teacher-student," where the large model is the teacher and the small model is the student.

The authors of YOLO provided multiple pretrained models with different backbone classifiers. Table 1 shows model size, performance, no. of parameters and layer depth of multiple backbone classifiers on the ImageNet validation benchmark. Among these classifiers, MobileNet-v2 has competing performance with very low memory and computational requirements. Hence, the teacher and student model we used is the pretrained YOLO-v3 (Redmon and Farhadi 2018) architecture with ResNet152 (He et al. 2016) backbone and the architecture shown in Fig. 2 with MobileNet-v2 (Sandler et al. 2018) backbone, respectively.

MobileNet-v2 is a CNN architecture developed by Sandler et al., 2018 particularly for mobile and embedded vision applications. The architecture uses depthwise separable convolution to reduce the model size and complexity. Table 2 shows the MobileNet body architecture. The following section discusses the training methodology and architecture of Mini-YOLO.

1, Dataset Used

We have used the Boxy vehicles dataset (Behrendt 2019) for training Mini-YOLO. Boxy vehicles dataset was created mainly for training vehicle detection algorithms in self-driving systems, hence most of the images from the dataset consists of vehicles on roads and traffic scenarios. This makes this dataset perfectly aligned for our use case as the input for our system is CCTV footages from roads and traffic scenarios.

The datasets contain 200,000 images of 1232 × 1028 resolution with 1,990,806 annotated vehicles by 3D-like and 2D bounding boxes. The 2D ground truth annotations from the dataset are used as hard labels during the distillation training procedure. Figure 3 shows some images along with the annotated 2D bounding boxes from the dataset.

2. Preprocessing

Each image in the dataset is down sampled to 512 × 512 resolution, and the custom annotation or soft labels for knowledge distillation is generated using the pretrained YOLO-v3 with ResNet152 backbone classifier. The prediction of the pretrained model (teacher model) is a 3D tensor of shape 19 × 19 × 85 which detects and classifies objects belonging to 80 different classes from the MS-COCO dataset.

Each output cell represents an area in the input image where a possible object bounding box center resides (19 × 19). Within each area, the model predicts the probability of presence of an object (1 × 1) along with probability of the object(if present) belonging to either of the 80 classes (1 × 80) and width, height, horizontal and vertical offsets (1 × 4) of the bounding box for the object detected giving the output tensor a shape 19 × 19 x (80 + 1 + 4) = 19 × 19 × 85.

For our system, there is only one class that needs to be detected viz "Vehicle". Hence, the output of our model (student model) is a 3D tensor of shape 19 × 19 × 6.

For generating soft labels for each image in the dataset, the predictions of only 4 classes, namely "motorcycle", "car", "truck" and "bus" are selected from the 80 available classes from the MS-COCO dataset. From these, the highest probable class prediction is taken and given the label of "Vehicle" to create a soft label annotation of shape 19 × 19 × 6. Figure 4 shows the flowchart for this procedure.

Table 1 Model size, performance, no. of parameters and depth of various backbone classifiers

Model	Size	Top-1 accuracy	Top-5 accuracy	Parameters	Depth
Xception	88 MB	0.790	0.945	22 M	126
VGG19	549 MB	0.713	0.900	143 M	26
ResNet50	98 MB	0.749	0.921	25 M	–
ResNet101	171 MB	0.764	0.928	44 M	–
ResNet152	232 MB	0.766	0.931	60 M	–
InceptionV3	92 MB	0.779	0.937	23 M	159
InceptionResNetV2	215 MB	0.803	0.953	55 M	572
MobileNet	16 MB	0.704	0.895	4 M	88
MobileNetV2	14 MB	0.713	0.901	3 M	88

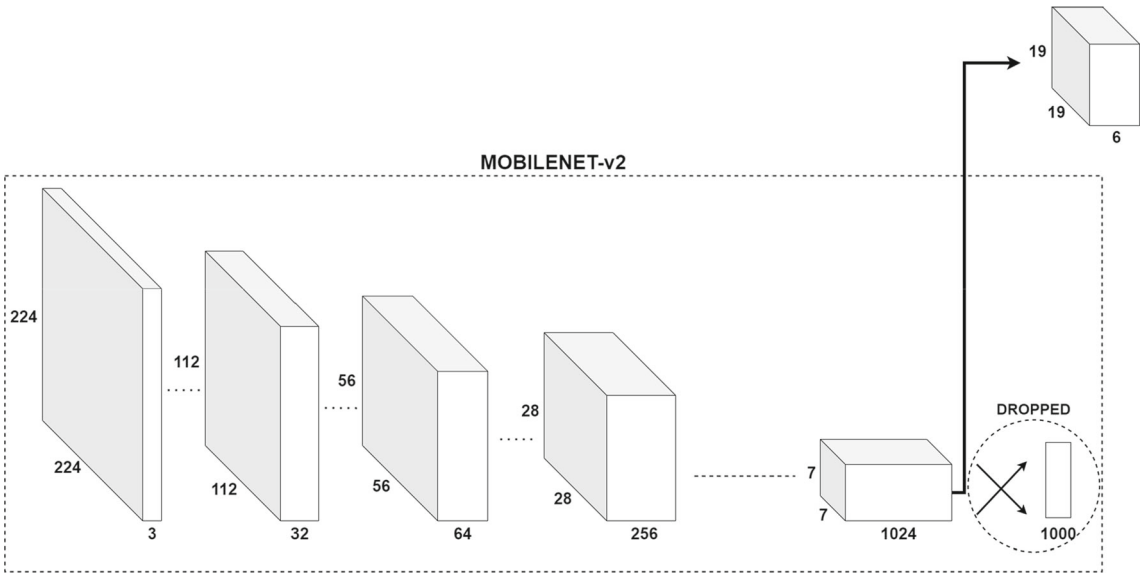

Fig. 2 Architecture for Mini-YOLO. The final classification layer of MobileNet-v2 is replaced with a 19 × 19 × 6 tensor

Table 2 Mobile net body architecture

Type/stride	Filter shape	Input size
Conv/s2	3 × 3 × 3 × 32	224 × 224 × 3
Conv dw/s1	3 × 3 × 32 dw	112 × 112 × 32
Conv/s1	1 × 1 × 32 × 64	112 × 112 × 32
Conv dw/s2	3 × 3 × 64 dw	112 × 112 × 64
Conv/s1	1 × 1 × 64 × 128	56 × 56 × 64
Conv dw/s1	3 × 3 × 128 dw	56 × 56 × 128
Conv/s1	1 × 1 × 128 × 128	56 × 56 × 128
Conv dw/s2	3 × 3 × 128 dw	56 × 56 × 128
Conv/s1	1 × 1 × 128 × 256	28 × 28 × 128
Conv dw/s1	3 × 3 × 256 dw	28 × 28 × 256
Conv/s1	1 × 1 × 256 × 256	28 × 28 × 256
Conv dw/s2	3 × 3 × 256 dw	28 × 28 × 256
Conv/s1	1 × 1 × 256 × 512	14 × 14 × 256
5 × Conv dw/s1		
Conv/s1	3 × 3 × 512 dw	
1 × 1 × 512 × 512	14 × 14 × 512	
14 × 14 × 512		
Conv dw/s2	3 × 3 × 512 dw	14 × 14 × 512
Conv/s1	1 × 1 × 512 × 1024	7 × 7 × 512
Conv dw/s2	3 × 3 × 1024 dw	7 × 7 × 1024
Conv/s1	1 × 1 × 1024 × 1024	7 × 7 × 1024
Avg Pool/s1	Pool 7 × 7	7 × 7 × 1024
FC/s1	1024 × 1000	1 × 1 × 1024
Softmax/s1	Classifier	1 × 1 × 1000

3. Model architecture

As a backbone classifier for our model, we have used the MobileNet-v2 architecture trained on the ImageNet dataset. Precisely, we replace the classification layer of the model with a 3D tensor of shape 19 × 19 × 6. Figure 2 shows the model architecture of proposed Mini-YOLO.

4. Training methodology

We train our model with knowledge distillation as proposed in Hinton et al., 2015. The authors of this paper have also suggested incorporating ground truth labels along with soft labels to increase model capacity and performance during training. Hence, we adopt the training strategy of Redmon et al., 2018 for training on the ground truth labels of the Boxy vehicles dataset.

By combining both the training strategies, the student model is able to attain knowledge from the MS-COCO (by learning to replicate the predictions of the pretrained teacher model) as well as the Boxy vehicles dataset (by learning to predict the ground truth annotations).

The training loss for ground truth predictions is a combination of 4 losses given by Eq. 1(a-c). Equation 1* & 2 provides the combined training loss for ground truth predictions and soft label predictions, respectively. Equation 3 represents the combined loss for training Mini-YOLO.

Equation 1(a): Classification loss .

If an object is detected, the classification loss at each cell is the squared error of the class conditional probabilities for each class:

$$L_1 = \sum_{i=0}^{s^2} 1_i^{\text{obj}} \sum_{c \in \text{classes}} (p_i(c) - \widehat{p}_i(c))^2 \qquad (1.a)$$

where $1_i^{\text{obj}} = 1$ if an object appears in cell i, otherwise 0.

$\widehat{p}_i(c)$ denotes the conditional class probability for class c in cell i.

Fig. 3 Some images along with the annotated 2D bounding boxes from the dataset

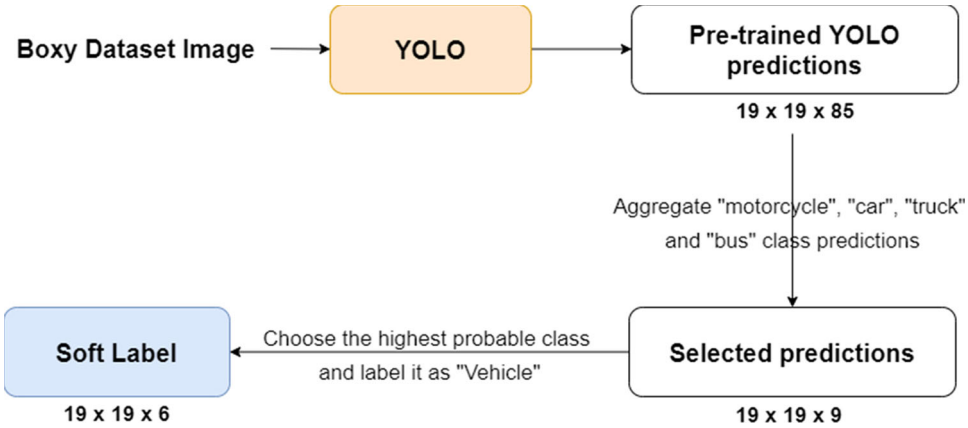

Fig. 4 Flowchart of Preprocessing for proposed Mini-YOLO

Equation 1(b): Localization loss.

The localization loss measures the errors in the predicted boundary box locations and sizes.

$$L_2 = \lambda_{\text{coord}} \sum_{i=0}^{s^2} \sum_{j=0}^{B} 1_{ij}^{\text{obj}} \left[(x_i - \hat{x}_i)^2 + (y_i - \hat{y}_i)^2 \right] + \lambda_{\text{coord}} \sum_{i=0}^{s^2} \sum_{j=0}^{B} 1_{ij}^{\text{obj}} \left[\left(\sqrt{w_i} - \sqrt{\hat{w}_i} \right)^2 + \left(\sqrt{h_i} - \sqrt{\hat{h}_i} \right)^2 \right] \tag{1.b}$$

where $1_{ij}^{\text{obj}} = 1$ if the j th boundary box in cell i is responsible for detecting the object, otherwise 0.

λ_{coord} increase the weight for the loss in the boundary box coordinates.

x, y are bounding box offsets.

w, h are bounding box width and height, respectively.

Equation 1(c): Confidence loss.

If an object is detected in the box, the confidence loss is as follows:

$$L_3 = \sum_{i=0}^{s^2} \sum_{j=0}^{B} 1_{ij}^{\text{obj}} \left(C_i - \widehat{C_i} \right)^2 \tag{1.1.c}$$

. where C_i is the box confidence score of the box j in cell i.

$1_{ij}^{\text{obj}} = 1$ if the j th boundary box in cell i is responsible for detecting the object, otherwise 0.

(1) If an object is not detected in the box, the confidence loss is as follows: $L_4 = \lambda_{\text{noobj}} \sum_{i=0}^{s^2} \sum_{j=0}^{B} 1_{ij}^{\text{noobj}} \left(C_i - \widehat{C_i}\right)^2$ (1.1.c)

where 1_{ij}^{noobj} is the complement of 1_{ij}^{obj}.

$\widehat{C_i}$ is the box confidence score of the box j in cell i. λ_{noobj} weights down the loss when detecting background.

Equation 1*: Combined training loss for ground truth predictions:

$$L_g = L_1 + L_2 + L_3 + L_4$$

Equation 2: Loss for soft label predictions:

$$L_s = \sum_{i=1}^{19} \sum_{j=1}^{19} \sum_{k=1}^{6} \left(y_{ijk} - \widehat{y_{ijk}}\right)^2 \quad (2)$$

where $\widehat{y_{ijk}}$ denotes the prediction of the teacher (YOLO) model.

y_{ijk} denotes the prediction of the student (Mini-YOLO) model.

Equation 3: Combined training loss for Mini-YOLO:

$$L = \alpha \times L_s + (1 - \alpha) \times L_g \quad (3)$$

where $\alpha \in [0, 1]$ is an hyperparameter for weighing soft label loss and ground truth loss.

5. Evaluation Metrics

For evaluating the detection performance of our model, we have computed AP score on the preprocessed MS-COCO dataset. AP or average precision score is a popular metric in measuring the accuracy of object detectors. Equation 4 provides the general definition of AP which is finding the area under the precision-recall curve.

Equation 4: General definition of average precision:

$$\text{AP} = \int_0^1 p(r)\mathrm{d}r \quad (4)$$

where $p(r)$ is precision for a given value of recall (r).

Along with reliable accuracy, the models must also be suitable for deployment in a real-time setting, hence, we evaluate our model performance by calculating the runtime in FPS (Frames per second) on a NVIDIA GTX 1060 MaxQ graphical processor. Our experiments with multiple backbone classifiers for detectors and other state-of-the-art detection models are provided in the result section.

2. Stage 2- Vehicle Tracking

Tracking vehicles and their damage status is a critical part in our system. Detecting accidents by just classifying each detected vehicle as damaged or not will not be a reliable mechanism in real life scenarios e.g., let's say a damaged vehicle is getting towed away from a location and this vehicle is detected by the system. Since it is damaged, the system will classify it as damaged with high confidence and this will trigger an accident false positive.

By keeping track of the detected vehicles, we can keep track of its damage status. This in turn will help in detecting whether a vehicle was brought damaged or was damaged during the transit, where the latter case is an accident true positive.

Keeping in mind the computational relaxation and task specificity (stationary cameras), we use a very simple object tracker "Simple Online and Realtime Tracking" proposed by Bewley et al., 2016.

SORT—Simple online and realtime tracking.

SORT explores a pragmatic approach to multiple objects tracking where the main focus is to associate objects efficiently for online and real-time applications. Despite only using a rudimentary combination of familiar techniques such as the Kalman Filter and Hungarian algorithm for the tracking components, this approach achieves an accuracy comparable to state-of-the-art online trackers. Furthermore, due to its simplicity, it achieves phenomenal runtime speed compared to other competing methods. In SORT, the authors considered the position and velocity of each of the object bounding boxes and used Kalman filtering to approximate these values from previously measured outcomes. After which, they used the Hungarian assignment algorithm to assign each of the bounding boxes to previously created tracks based on the Euclidean distance between them or initialize a new track if the distance is more than a specified threshold.

SORT was originally proposed as a person tracking framework but as it did not account for the appearance factor of the object, we extend it to our use case. Each acquired frame from the camera is inferenced across Mini-YOLO for vehicle bounding boxes. These boxes are then passed on to SORT for associating it with a previously detected Vehicle or to consider as a new one. Each track initialized is assigned a damage status variable which is updated by the next stage, the accident classification stage.

3. Stage 3- Accident classification

Once vehicles are detected from the input frame, the images of detections are extracted using the bounding box and each of those images are classified for damage status. Higher damage status indicates damaged vehicle triggering an accident.

We have trained an SVM with radial basis kernel to classify detected vehicle images for damage status. The following section discusses the training methodology in detail.

1. Dataset Used

Accident-Images-Analysis-Dataset (Pashaei et al. 2019) has been used for the training of the classification model. The dataset contains a total of 10,480 images of damaged and un-damaged vehicles. Most of the images in the database used were directly scraped from the web. As a result, a larger portion consists of high-quality images. Figure 5 shows some images along with their labels from the dataset.

2. Preprocessing

Each image in the dataset is down sampled to 224 × 224 resolution and are converted to grayscale color space. As most of the images in the database were directly scraped from the web, a larger portion consisted of high-quality images. To increase generalization of the model in deployment, we have added nominal Gaussian noise to mimic inputs taken directly from low resolution CCTVs. Such a procedure has shown significant increase in accuracy at inference time.

3. Training methodology

We trained a support vector machine with radial basis kernel for classifying vehicle images into damaged and undamaged categories. The most important parameter of SVM is slackness or C, we have set this parameter to 1.0 while the value of gamma for the radial basis function is set to 0.1 which is the default value in scikit-learn, the Python software library that we used to train the SVM.

4. Evaluation metrics

For evaluating the classification performance of the SVM classifier, we have computed the precision and recall on the preprocessed Accident-Images-Analysis-Dataset. Precision and recall are two extremely important model evaluation metrics. Precision is the fraction of correctly classified positive instances among the total number of instances classified as positive, while recall is the fraction of the total number of positive class instances that were correctly classified. Equation 5 provides the formula for calculating precision and recall. We also compare our model with multiple algorithms in terms of latency (in ms) and size. These experiments along with results are highlighted in the result section.

Equation 5(a): Precision:

$$\text{Precision} = \frac{\text{True Positive}}{\text{True Positive} + \text{False Positive}}$$

where True Positive is the total number of positive instances correctly classified.

False Positive is the total number of negative instances incorrectly classified as positive.

Equation 5(b): Recall:

$$\text{Recall} = \frac{\text{True Positive}}{\text{True Positive} + \text{False Negative}}$$

where True Positive is the total number of positive instances correctly classified.

False Negative is the total number of positive instances incorrectly classified as negative.

The detailed flowchart of all 3 components connected together is shown in Fig. 6. First, the image acquired from the input camera is inferenced across Mini-YOLO for vehicle bounding boxes. Then, each detected bounding box is passed onto SORT for track association and the associated tracks are initialized with a damage indicator variable.

Fig. 5 Some images along with their labels from the Accident-Image-Dataset. Label 1 is for damaged vehicles, and label 0 is for undamaged ones

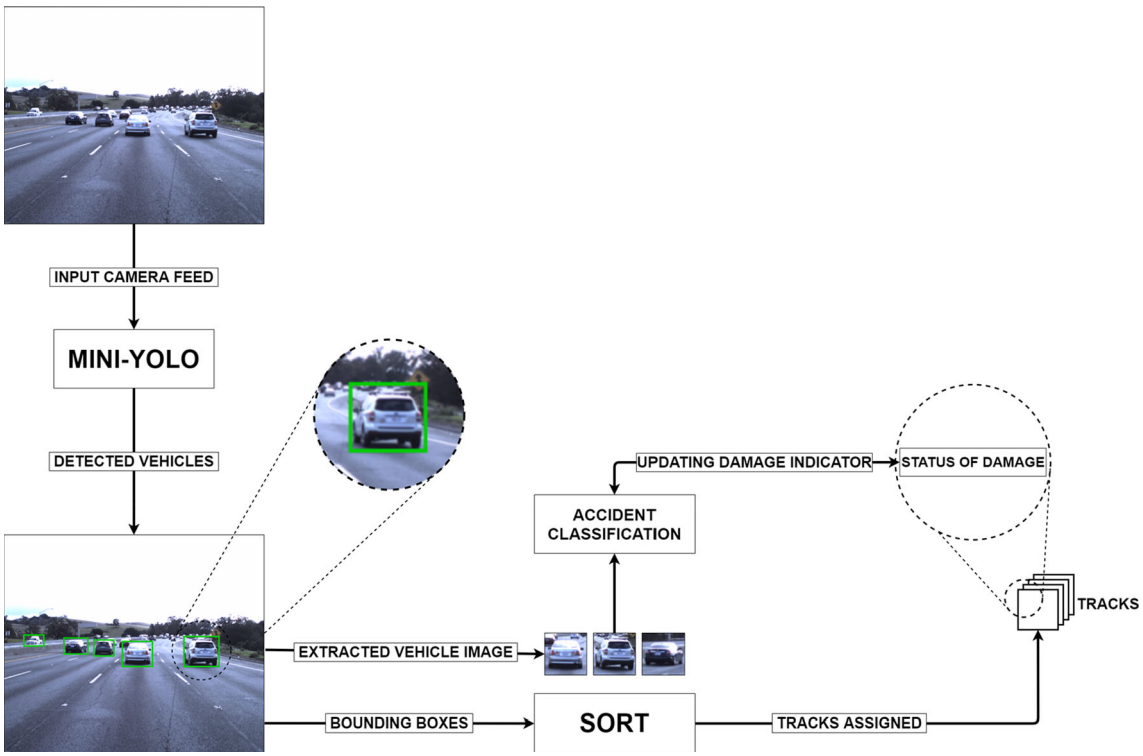

Fig. 6 Flowchart of the proposed system

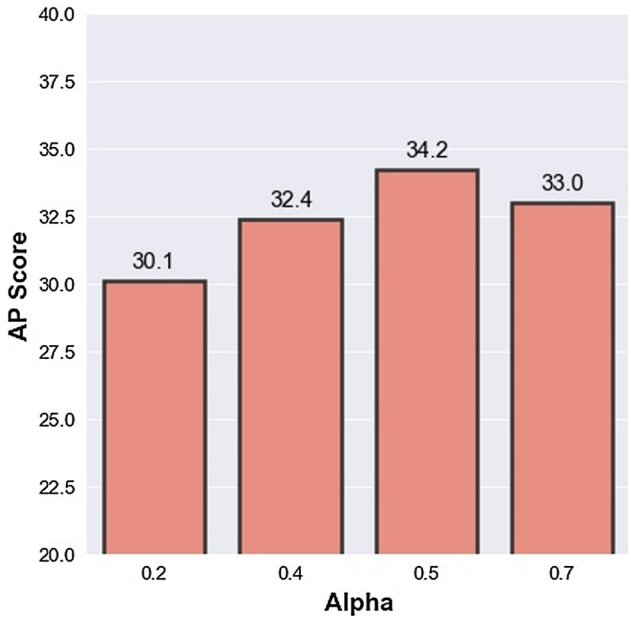

Fig. 7 The AP scores obtained for multiple values of α

This variable is controlled by the accident classification system which takes the extracted vehicle image (using the bounding box) and classifies each detected vehicle into damaged and undamaged categories. If a vehicle is classified as damaged and the damage indicator variable of the associated track is set to false (not damaged from the start) then the variable is set to true (damaged during transit) and an accident alarm is raised.

Table 3 Performance comparison of multiple detection algorithms

Model	Backbone classifier	AP score	Runtime (FPS)
Faster R-CNN	Inception-ResNet-v2	34.7	5
SSD	ResNet-101	31.2	9
YOLOv3	ResNet-101	34.9	17
	ResNet-152	36.2	12
Mini-YOLO	MobileNet-v2	34.2	28

3 Results

The performance of the proposed Mini-YOLO is compared with other detection algorithms with the metrics provided in Sect. 2.1.5. Precisely for comparison, we adopted Faster R-CNN with Inception-ResNet-v2 (Ren et al. 2015) backbone, SSD with ResNet-101 (Liu et al. 2016) backbone and YOLOv3 with ResNet-101 and ResNet-152 backbone classifiers based on their computational requirements and ease of implementation. We trained these architectures

Table 4 Performance of the accident classification model

Model	Precision	Recall	F1-score	AUROC score	Model size	Latency
Decision tree	0.79	0.78	0.78	0.89	192 KB	12.92 ms
CNN	0.97	0.98	0.97	0.98	29.9 MB	76.61 ms
Random forest	0.82	0.90	0.85	0.93	704 KB	25.47 ms
SVM-linear	0.80	0.76	0.77	0.84	128 KB	12.73 ms
SVM-rbf	0.96sssss	0.94	0.95	0.96	448 KB	12.73 ms

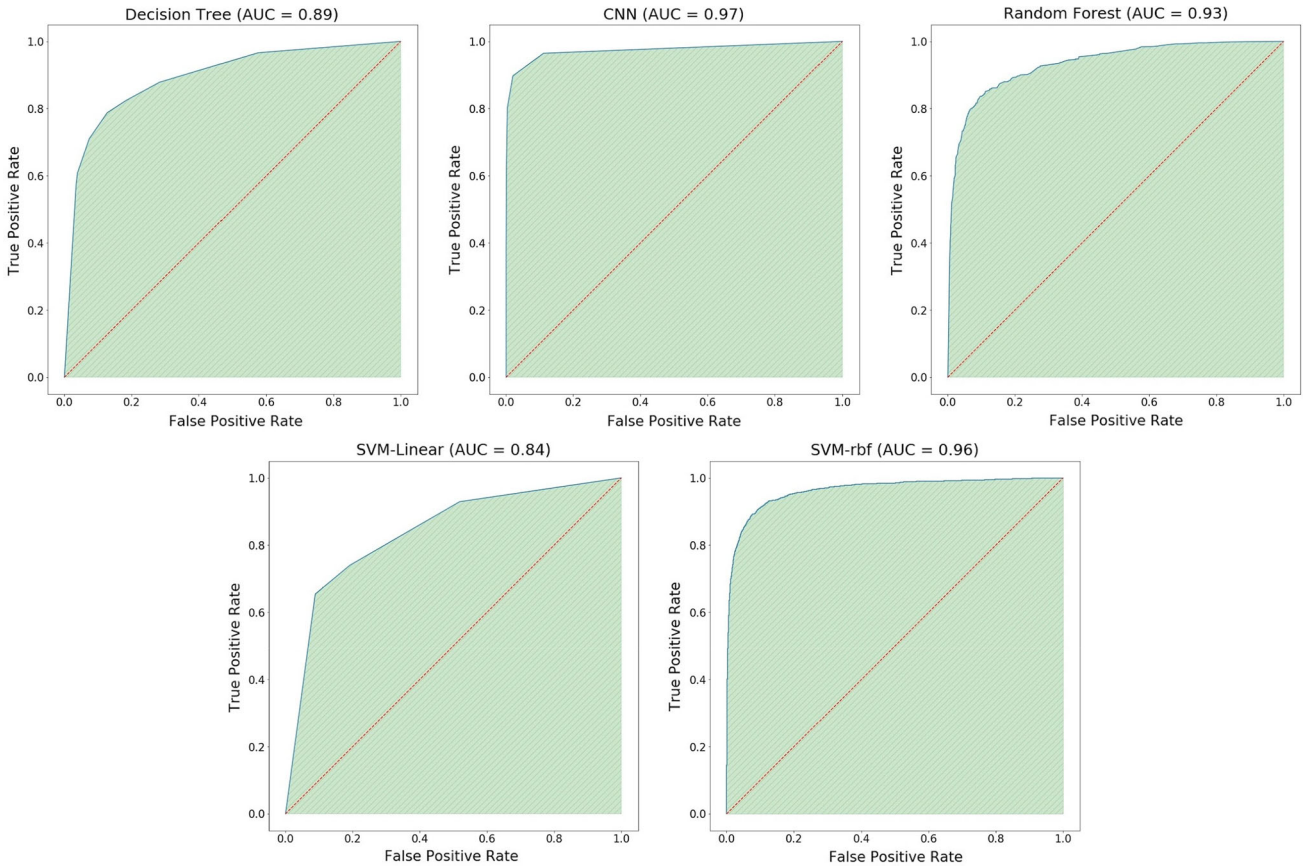

Fig. 8 Receiver operating characteristic curve of different classification model

along with our proposed Mini-YOLO on the MS-COCO dataset and calculated the average precision (AP) score on its testing benchmark. Each of these algorithms were further evaluated on the basis of runtime (in FPS).

For training Mini-YOLO with the combined training loss, we experimented with $\alpha\{0.2, 0.4, 0.5, 0.7\}$ and found out that for our settings $\alpha = 0.5$ obtains the highest AP score. Figure 7 shows the AP scores obtained for multiple values of α.

The runtime and AP score of multiple detection algorithms with the best AP score of Mini-YOLO is shown in Table 3. It can be easily inferred that our proposed Mini-YOLO outperforms all other detection models in terms of runtime while keeping high comparable accuracy at a mere size of 18 MB.

We experimented with multiple classification algorithms to employ in the accident classification stage. Namely, we trained a decision tree classifier, a convolutional neural network (CNN) classifier, a random forest classifier and two support vector machine (SVM) classifiers with linear and radial basis function (RBF) kernels on the Accident-Images-Analysis-Dataset.[1] These algorithms were then evaluated on the testing set with the metrics provided in Sect. 2.3.4 along with the AUROC (Area under Receiver Operating Characteristic curve) score. Table 4 shows the calculated precision, recall and AUROC score of the

[1] These algorithms were adopted based on their computational inexpensiveness and memory requirements (except CNN). We trained the CNN to only produce a high-order comparable result on the AUROC score.

algorithms. The metric "Model Size" is the size of the model (in KB/MB) file loaded in the random-access memory (RAM) of our system and "Latency" is the amount of time taken (in millisecond) to classify a single input image. It can be inferred from the results that an SVM classifier with RBF kernel (SVM-rbf) works best for our case with very low latency and high-order classification score. Figure 8 shows the receiver operating characteristic (ROC) curve of this model and the shaded green area represents the area under the curve (AUC).

4 Conclusion

In this paper, we have successfully applied a computationally traceable and reliable automatic accident detection system that can be deployed in scenarios requiring real-time feedback. We also presented Mini-YOLO; a deep learning architecture trained using knowledge distillation that has phenomenal runtime speed with a high AP score compared to other detection algorithms. For accident classification, we experimented with multiple algorithms and showed that a support vector machine with radial basis kernel performs the best in terms of memory requirement and latency.

5 Limitation and future work

Given the confidence level in the results obtained, there is still scope for field testing and improvements with respect to deployment and maintenance. Also, as we have only incorporated the static notion of accidents, viz damaged vehicles, we believe that the performance of the proposed accident classification algorithm could further be improved through the use of motion models and sequence processing. This will not only increase the classification robustness but will also reduce the necessity of the tracking stage. Therefore, our main concern in the near future is to extend the accident classification stage to incorporate sequence models in a computationally inexpensive manner.

Funding This study was no funding.

Compliance with ethical standards

Conflict of interest There is no conflict of interest.

Ethical approval This article does not contain any studies with human participants or animals performed by any of the authors.

References

Ahangari S, Jeihani M, Dehzangi A (2019) A machine learning distracted driving prediction model. In: International symposium of intelligent unmanned systems on artificial intelligence.

Alwan ZS, Muhammed H, Alshaibani A (2016) Car accident detection and notification system using smartphone. Int. J. Comput. Sci. Mob. Comput.(January).

Arvin R, Kamrani M, Khattak AJ (2019) How instantaneous driving behavior contributes to crashes at intersections: extracting useful information from connected vehicle message data. Accid Anal Prev 127:118–133

Azimi G, Asgari H, Rahimi A, Xia J (2019) Investigation of heterogeneity in severity analysis for large truck crashes. In: 98th Annu. Meet. Transp. Res. Board.

Behrendt K (2019) Boxy vehicle detection in large images. In: Proceedings of the IEEE international conference on computer vision workshops (pp. 0–0).

Bewley A, Ge Z, Ott L, Ramos F, Upcroft B (2016) Simple online and realtime tracking. In: 2016 IEEE international conference on image processing (ICIP) (pp. 3464–3468). IEEE.

Buciluǎ C, Caruana R, Niculescu-Mizil A (2006) Model compression. In: Proceedings of the 12th ACM SIGKDD international conference on knowledge discovery and data mining (pp. 535–541).

Chen Y, Yu Y, Li T (2016b) A vision based traffic accident detection method using extreme learning machine. In: Int. Conf. Adv. Robot. Mechatronics 567–572. https://doi.org/10.1109/ICARM.2016.7606983.

Dong N, Huang H, Zheng L (2015) Support vector machine in crash prediction at the level of traffic analysis zones: assessing the spatial proximity effects. Accid Anal Prev 82:192–198. https://doi.org/10.1016/j.aap.2015.05.018

Fernandes B, Alam M, Gomes V, Ferreira J, Oliveira A (2016) Automatic accident detection with multi-modal alert system implementation for ITS. Veh Commun 3:1–11. https://doi.org/10.1016/j.vehcom.2015.11.001

Global status report on road safety (2015) 2015. World Heal, Organ

Golshani N, Shabanpour R, Mahmoudifard SM, Derrible S, Mohammadian A (2018) Modeling travel mode and timing decisions: comparison of artificial neural networks and copula-based joint model. Travel Behav. Soc. (September 2017), 21–32. https://doi.org/10.1016/j.tbs.2017.09.003.

Gu Y, Sean Z, Chen F (2016) From Twitter to detector: real-time traffic incident detection using social media data. Transp Res Part C 67:321–342. https://doi.org/10.1016/j.trc.2016.02.011

He K, Zhang X, Ren S, Sun J (2016) Deep residual learning for image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 770–778).

Hinton G, Vinyals O, Dean J (2015) Distilling the knowledge in a neural network. arXiv preprint arXiv:1503.02531.

Kumar KHM, Harish BS (2019) Automatic irony detection using feature fusion and ensemble classifier. Int J Interact Multimed Artif Intell 5(7):70–79

Lin TY, Maire M, Belongie S, Hays J, Perona P, Ramanan D, Dollár P, Zitnick CL (2014) Microsoft coco: common objects in context. In: European conference on computer vision (pp. 740–755). Springer, Cham.

Liu W, Anguelov D, Erhan D, Szegedy C, Reed S, Fu CY, Berg AC (2016) Ssd: Single shot multibox detector. In: European conference on computer vision (pp. 21–37). Springer, Cham.

Maaloul B, Taleb-ahmed A, Niar S, Harb N, Valderrama C (2017) Adaptive videobased algorithm for accident detection on highways. In: 12th IEEE Int. Symp. Ind. Embed. Syst. 1–6. https://doi.org/10.1109/SIES.2017.7993382.

Traffic Incident Management (2013) Fed. Highw. Adm. Traffic safety facts 2012: Young drivers,2014. Natl. Highw. Traffic Saf. Adm

Mansourkhaki A, Karimpour A, Sadoghi Yazdi H (2016) Non-stationary concept of accident prediction. Proc Inst Civil Eng Transp 170:140–151

Mansourkhaki A, Karimpour A, Sadoghi Yazdi H (2017) Introducing prior knowledge for a hybrid accident prediction model. KSCE J Civil Eng 21(5):1912–1918

Mokhtarimousavi S, Anderson JC, Azizinamini A, Hadi M (2019) Improved support vector machine models for work zone crash injury severity prediction and analysis. Res. Rec, Transp. https://doi.org/10.1177/0361198119845899

Nasr Esfahani H, Arvin R, Song Z, Sze NN (2019) Prevalence of cell phone use while driving and its impact on driving performance, focusing on near-crash risk: a survey study in Tehran. J Transp Safety Secur. https://doi.org/10.1080/19439962.2019.1701166

Naz S, Ziauddin S, Shahid AR (2019) Driver fatigue detection using mean intensity, SVM, and SIFT, Int J Interact Multimed. Artif Intell 5(4):86–93

Ozbayoglu M, Kucukayan G, Dogdu E (2017) A Real-time autonomous highway accident detection model based on big data processing and computational intelligence. IEEE, pp. 1807–1813.

Parsa AB, Taghipour H, Derrible S, Mohammadian AK (2019) Real-time accident detection: coping with imbalanced data. Accid Anal Prev 129(January):202–210. https://doi.org/10.1016/j.aap.2019.05.014

Parsa AB, Kamal K, Taghipour H, Mohammadian AK (2019b) Does security of neighborhoods affect non-mandatory trips? a copula-based joint multinomial-ordinal model of mode and trip distance choices. In: Transportation Research Board 98th Annual Meeting.

Parsa AB, Chauhan RS, Taghipour H, Derrible S, Mohammadian A (Kouros) (2019a) Applying deep learning to detect traffic accidents in real time using spatiotemporal sequential data. arXiv Preprint arXiv 1912.06991.

Pashaei A, Ghatee M, Sajedi H (2019) Convolution neural network joint with mixture of extreme learning machines for feature extraction and classification of accident images. J Real-Time Image Process 17(4):1–16

Razi-Ardakani H, Mahmoudzadeh A, Kermanshah M (2018) A nested logit analysis of the influence of distraction on types of vehicle crashes. Eur Transp Res Rev 10:44

Redmon J, Divvala S, Girshick R, Farhadi A (2016) You only look once: unified, real-time object detection. In: Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 779–788).

Redmon J, Farhadi A (2018) Yolov3: an incremental improvement. arXiv preprint arXiv:1804.02767.

Ren S, He K, Girshick R, Sun J (2015) Faster r-cnn: towards real-time object detection with region proposal networks. In: Advances in neural information processing systems (pp. 91–99).

Sandler M, Howard A, Zhu M, Zhmoginov A, Chen LC (2018) Mobilenetv2: inverted residuals and linear bottlenecks. In: Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 4510–4520).

Sharifi MS, Song Z, Nasr Esfahani H, Christensen K (2019) Exploring heterogeneous pedestrian stream characteristics at walking facilities with different angle intersections. Phys A: Stat Mech Appl 540:123112

Sun J, Sun J (2015) A dynamic Bayesian network model for real-time crash prediction using traffic speed conditions data. Transp Res Part C 54:176–186. https://doi.org/10.1016/j.trc.2015.03.006

Thomas RW, Vidal JM (2017) Toward detecting accidents with already available passive traffic information. In: IEEE 7th Annu. Comput. Commun. Work. Conf. 1–4. https://doi.org/10.1109/CCWC.2017.7868428.

Traffic Safety Facts FARS, 2013. Natl. Highw. Traffic Saf. Adm. (NHTSA), GES Annu. Rep.Natl. Highw. Traffic Saf. Adm. (NHTSA), GES Annu. Rep.

Vishnu VCM, Nedunchezhian MRR (2018) Intelligent traffic video surveillance and accident detection system with dynamic traffic signal control. Cluster Comput 21(1):135–147. https://doi.org/10.1007/s10586-017-0974-5

Xia S, Xiong J, Liu Y, Li G (2015) Vision-based traffic accident detection using matrix approximation. In: 10th Asian Control Conf 1–5. https://doi.org/10.1109/ASCC.2015.7244586.

Zaldivar J, Calafate CT, Cano JC, Manzoni P (2011) Providing accident detection in vehicular networks through OBD-II devices and android-based smartphones. In: IEEE 36th Conf. Local Comput. Networks. IEEE. pp. 813–819.