Deep Learning Envisioned Accident Detection System

Intekhab Alam, Ayush Verma, and Manju Khari(✉)

Jawaharlal Nehru University, New Delhi 110067, India
{intekh31_scs,ayushv34_scs}@jnu.ac.in, manjukhari@yahoo.co.in

Abstract. The conflict between computational overhead and detection accuracy affects nearly every Automated Accident Detection (AAD) system. Although the accuracy of detection and classification approaches has recently improved dramatically, the systems' high computational resource requirements make them unsuitable for deployment in situations when immediate feedback is required. This paper suggests a strategy for developing an accurate, cost-efficient automatic collision identification system that can be deployed with a minimal amount of hardware. The AAD system is divided into three key phases—Detection, Tracking, and Classification and offers computationally reduced intensive algorithms for each of these stages. For the detection phase, YOLOv3, it uses a deep learning model that has been trained through knowledge distillation. Its accuracy is on par with that of YOLO (You-Only-Look-Once). On the basis of the information inference and data mining techniques COCO (Common Objects in Context) dataset, YOLOv3 surpasses all other identification techniques considering runtime complexity, averaging an astounding 30 frames/second on a low-end system. It attains an Average Precision (AP) mark of 44.8. For the tracking phase, the system makes use of SORT (Simple Online Real-time Tracking). Since a radial basis kernel with a support vector machine works efficiently, an area under the curve (or AUC) score of 0.92 is achieved. Alongside this, this paper examines a number of machine-learning techniques for the classification stage.

Keywords: Automatic Accident Detection · Computer Vision · Deep Learning · YOLO · Vehicle Tracking

1 Introduction

Moving both individuals and possessions from one place to another requires efficient and effective transportation in today's world. However, there are also major risks associated with transportation, most notably in terms of accidents. Traffic accidents cause millions of deaths and injuries each year, posing a severe threat to global public health. Traffic accidents have a substantial social and economic impact as well, costing an estimated $518 billion annually throughout the globe [1]. The causes of traffic accidents are complex and multifaceted, involving a combination of human, environmental, and technological factors. Human factors, such as driver behavior and error, are the most common cause of accidents, accounting for up to 90% of all accidents [2]. Accidents can also be caused by environmental variables, like as weather and road design. Technological factors, such as vehicle design and safety features, can help mitigate the risk of accidents but can also introduce new risks, such as distracted driving. To address the issue of traffic accidents, researchers and engineers have been exploring various approaches to develop intelligent transportation systems that can detect and prevent accidents in real-time. These systems can help reduce the number of accidents and improve overall road safety, ultimately leading to more efficient and sustainable transportation systems. Advanced technologies like artificial intelligence and computer vision enable the development of intelligent solutions capable of real-time accident detection and prevention, ultimately saving lives and improving the overall efficiency of our transportation systems. In addition to the human and economic costs of traffic accidents, there are also broader societal implications to consider.

Traffic accidents can lead to significant disruptions in transportation systems, causing delays and inefficiencies that can have a ripple effect on other aspects of society, such as commerce and healthcare. Additionally, accidents involving vehicles can harm the environment, particularly through air pollution and greenhouse gas emissions. To address these broader societal implications, it is essential to take a holistic approach to transportation and accident prevention. This approach should involve a combination of technological solutions, such as intelligent transportation systems, as well as policy and regulatory measures, such as improved road safety regulations and enforcement of traffic laws. Additionally, public education and awareness campaigns can help raise understanding about the risks of traffic accidents and encourage safer driving behaviors. In conclusion, transportation and accidents are inextricably linked, with traffic accidents posing significant risks to public health, the economy, and the environment. By investing in research and development in this field, we can develop innovative solutions that can help reduce the number of accidents and improve overall road safety, ultimately leading to more efficient and sustainable transportation systems. Additionally, by taking a holistic approach to transportation and accident prevention, we can address the broader societal implications of traffic accidents and create a safer, more sustainable future for all.

The rest of this paper is organized as follows: Sect. 2 discusses about literature survey and previous works related to accident detection systems. Section 3 explains the methodology followed for implementation of algorithms. Section 4 discusses the results and validation measures for comparison among various algorithms. Section 5 concludes the paper with overview of the paper and future directions.

2 Literature Survey

Kapri et al. [3] devised a system which states that in a remote location when no one is present for reporting it, an accident could happen. High-end vehicles now come equipped with hardware elements that may detect and record accidents. Unfortunately, these gadgets are immovable and pricey.

A system using HDy Copilot, an Android app, was recommended by Fernandes et al. [4] for detecting accidents related to multimodal alert dissemination. The user can also access the app, get notifications from nearby vehicles for potential road dangers, and stop the countdown timer if a false collision is discovered by using an android phone, which is also used to access the app. When a car crash happens, an alert is sent out to all nearby vehicles, and texts and phone calls are also made to the rescue number. This is demonstrated on the e-Call platform. However, the core problem—phone breakage or signal loss—remains unchanged, making the gadget ineffective.

Piedad et al. [5] analysis is based on Hungarian algorithm information inference and data mining techniques. It developed a paradigm that outlines a smart system that detects traffic problems across vehicle networks quickly, warns of them, and evaluates their seriousness. The authors have created and used a prototype towards automated accidents. After a careful examination of all pertinent characteristics, the results demonstrated that a vehicle's speed is a significant contributing element in front collisions. The explored classification algorithms do not significantly differ from one another, and they do demonstrate that the efficiency of the model can be markedly improved unless accidents are classed according to the impacts, especially in front crashes in which the vehicle is usually the one that strikes.

According to Priyanka et al. [6] description of this technique, aims to avoid accidents by identifying the driver's circumstances, such as when they are too tired or unfit to drive. Infrared Radiation reflecting obstacle sensors are used to determine the positioning of the eyes. The sensor transmits to the controller a weak signal as soon as the eyes open, and a high signal after the eyes have been closed for at least four seconds. Input from the sensor is processed by the PIC controller to turn on or off the alarm system and, if the alarm sound is ON, display "Driver Slept" on the display. The low or high signal is processed by the controller. The low or high signal is processed by the controller. As therefore, driver is informed possibly avoiding a serious collision. Since fatigue plays a large role in accidents, this kind of technology offers an especially efficient way of preventing such occurrences. Although the system offers a means of preventing accidents brought on by drunkenness, the fact that it does not offer damage detection means that there may be further reasons that are not discovered and the emergency contacts are not informed would be a serious drawback.

According to Bewley et al. [7], the primary goal of this study's measured approach of multiple item detection and tracking is to effectively connect objects for usage in actual time as well as online applications. For this reason, it is shown that tracking performance is greatly influenced by detection quality and that improving the algorithm can improve tracking up to 18.9%. Even though the tracking components of this system are only tracked using a straightforward combination of well-known methods, which include the Kalman Filter with Hungarian algorithm, it nonetheless achieves tracking accuracy on par with other online trackers. The tracker changes at an average rate of 260 Hz, which is around 20x faster with respect to other cutting-edge trackers.

Dogru et al. [8] proposed system broadcasts traffic alerts to the drivers using simulated data obtained from vehicular ad-hoc network (VANETs) relying on the velocity as well as position of the cars. Additionally, it demonstrates how ITS may use machine learning methods to detect collisions on freeways. In order to develop a model that can identify between accident cases and usual cases, supervised machine learning methods like random forest models (RF), Artificial Neural Networks (ANN), and support vector machine are used to traffic data. In terms of accuracy, the RF algorithm was found to perform superior to the ANN and SVM algorithms. With 91.56% accuracy, the RF algorithm performed better than the SVM (88.71%) and the ANN (90.02%) algorithms.

Chand et al. [9] presented a cutting-edge support system for self-driving automobiles that can detect collisions using a dashboard camera. The system uses the centroid tracker technique for recognizing the observed vehicle and the Mask R-CNN framework to detect cars. In order to determine whether an accident has taken place amongst all of the recognized vehicles, the approach additionally computes a number of factors, including as speed, momentum, and trajectory.

Tsuji et al. [10] address the construction of an extensive pedestrian detecting system. They discuss the placement of the cameras, coordinate systems, simple IR-based recognition, surveillance, computing relative movement vectors, and collision evaluation standards.

Wakim et al. [11] propose a Markovian hypothesis to explain pedestrian behavior. Using a hidden Markov model, the dynamics of pedestrians are represented by four states: sitting still, strolling, jogging, and running. The likelihood of changing states is shown on a model transition diagram. The probability distributions of absolute speed and change in direction were modeled as truncated Gaussians for each state. Collision probabilities are forecasted using Monte-Carlo simulations.

In their Probabilistic framework for multi-object tracking, Antonini et al. [12] used this setup to analyze pedestrian trajectory and assign marks for doing so. The way a pedestrian's body is oriented frequently reveals important details about the direction of movement to come. Therefore, determining the pedestrian orientation may help with motion prediction and provide more accurate collision probability predictions. Support Vector Machines based on Hartley wavelet coefficients are used by Poggio et al. [13] to estimate a person's orientation and classify amongst various orientations. To extract the automobile's characteristics (velocity, momentum, and direction), Zu et al. [14] use the Mean Shift algorithms for tracking as well as a motion detection using the Gaussian Mixture Model (GMM). Ren et al. [15] suggestion is to first extract segments of road and line positions before applying background subtraction for obtaining moving objects. Then a time and space graphic with a traffic tracking graph is produced. For classification and accident recognition, a Support Vector Machine (SVM) was employed.

3 Methodology

Our system is designed to automate incident detection in real-time using a three-phase process: detection, tracking, and classification. In the detection phase, we employ the You Only Look Once (YOLO) method [16], a popular deep learning algorithm, to identify and categorize moving cars in images and video frames. YOLO divides the input into a grid-like arrangement of cells and predicts box boundaries and class likelihoods for each cell. This approach is faster than traditional object detection algorithms and allows for the detection of multiple items in a single pass. The YOLO method has proven to be versatile and can be applied to various tasks beyond accident detection, including traffic monitoring, vehicle counting, and license plate recognition [17]. The data generated by these systems can be valuable for traffic management and law enforcement organizations. To track the detected cars over time, we utilize the Simple Online and Realtime Tracking (SORT) algorithm. SORT combines well with YOLO, as it focuses on object tracking while YOLO excels at object identification and categorization. Studies have demonstrated that the YOLO and SORT algorithms achieve high accuracy and reliability in object detection and tracking [18]. In the classification stage, we evaluate different machine learning techniques to classify the segmented automobile images from the detection phase into damaged and undamaged classes. This step allows us to assess the state of damage for each detected car. By employing less computationally intensive algorithms for each stage, we aim to create a reliable and economically affordable real-time incident alert system that requires minimal equipment. In addition to accident detection, other related works have explored the application of deep learning techniques for driver safety. For instance, a real-time driver drowsiness detection model based on facial features has been proposed [19]. This system quickly detects signs of drowsiness in drivers and provides warnings to ensure their safety. The YOLO developers have provided a range of trained models with different backbone classifiers. Table 1 showcases the model size, performance, parameter quantity, and layer depth for several backbone classifiers on the ImageNet validation benchmark. Among these classifiers, MobileNet-v2 stands out for its competitive performance and minimal memory and processing requirements. Therefore, we have employed the pretrained YOLO-v3 model, which utilizes MobileNet-v2 as the backbone, for both the teacher and student models. Figure 1 represents the top-level design of the proposed methodology.

Table 1. Multiple backbone classifier model's depth, parameter count and size.

Classifier	Size	Parameter	Depth	Accuracy
VGG19	543 MB	143 M	26	72%
ResNet50	100 MB	25 M	20	74.6%
ResNet101	175 MB	44 M	20	76.9%
MobileNet	16 MB	4 M	90	71%
MobileNetV2	14 MB	3 M	90	72.5%

3.1 Phase 1 - Vehicle Detection

In view of the necessity for significant recall and computing lightness, we used the YOLO v3 vehicle recognition method, which is computationally light and has a high recall. YOLO v3 is ahead of the pack for a number of reasons, including its:

- Quickness
- Detection precision
- Excellent generalization
- Freely available

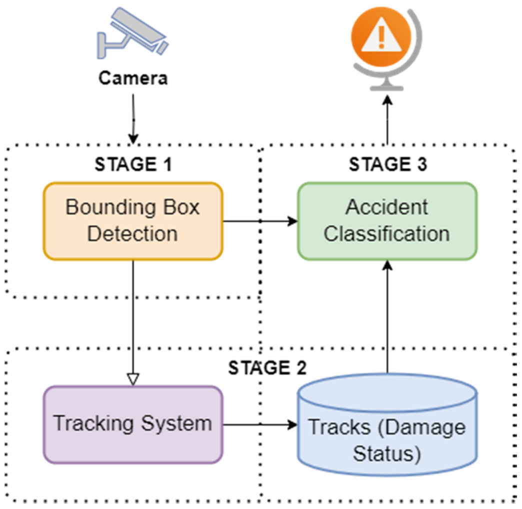

Fig. 1. Top-level design of the suggested system

Implementing the following four methods, the algorithm operates:

1. **Grid blocks:** In this first step, uniformly shaped N × N cell grids are created from the original image (A). The class of every grid cell and the confidence value of probability value should be predicted locally. This is done along with the coverage of the item of grid cell.
2. **Bounding box:** The next step is to localize the borders of the bounding box. These boxes are the rectangles that cover/bound every article present in the image. It is possible to have several box boundaries as there are items in each image. All the final bounding boxes are stored in Y which is the final matrix/vector form of each bounding box.

$$Y = [px, bc, bd, bh, bw, C1, C2] \tag{1}$$

This is especially important when the model is being trained.

- The value of *px* denotes the object's square's likelihood value.
- The x and y location coordinates of the center of the bounding box with respect to the adjacent grid cell - *bc* and *bd*.
- The letters *bh* and *bw* stand for the bounded box's height and breadth in reference to the adjacent grid cell.
- The classes *C1* and *C2* serve as substitutes for Player or Ball, respectively. More classes are available if your use cases require them.

IOU (Intersection Over Union): The IOU's (a value within 0 and 1) purpose is to eliminate such grid boxes and retain only the necessary ones. Here is the reasoning for it:

The IOU selection threshold is set by the user and can be, for example, 0.5.

The IOU, which is the the intersection Region divide by the Union Area, is then calculated for each grid cell using YOLO.

Finally, it considers grid cells with an an IOU > limit rather than those predicted to have an IOU threshold.

Non-Maximum Suppression: Setting an IOU ceiling isn't always enough because an item could have many boxes with IOU that's over the limit, and if all of those boxes are left open, noise might get in. It can be used in this situation to conserve only the items having the best chance of being found.

Dataset

For YOLOv3 training, made use of the Boxy Automobiles Dataset [20]. Since the primary purpose of the Boxy cars dataset was to train vehicle identification algorithm in self-driving systems, the majority of the dataset's photos include automobiles in traffic and on roads. Given that our system receives CCTV footage from scenes of busy highways and traffic, for our use case, this dataset works out exactly right. This files include 200,000 1232 × 1028-pixel photos with 1,990,806 automobiles tagged with 2D and 3D bounding rectangles. During the distillation training phase, annotated images (ground truth) which are two-dimensional are used as hard labels. Figure 2 represents a sample image of the dataset which is used for YOLOv3 training.

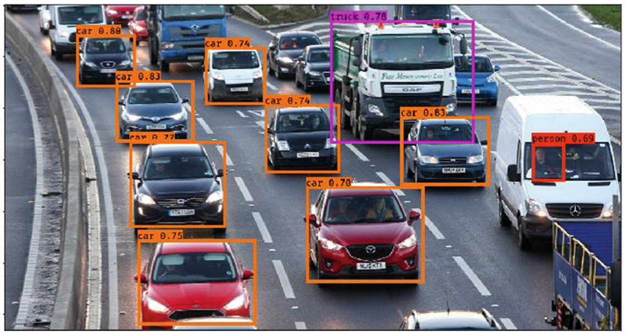

Fig. 2. Two-dimensional bounding boxes with annotations

Preprocessing

The dataset's images are all down scaled to 512 × 512 resolution, and the trained YOLO v3 using ResNet152 backbone classifier is used to generate the custom annotation or soft labels. A three-dimensional tensor with the shape 19*19*85 that recognizes and categorizes items from the MS-COCO dataset that correspond to 80 different classes is predicted of the trained model (teacher model). Here only exists one class, "Vehicle," that has to be identified by our system. As a result, our model's (the student model's) output is a 3D tensor with the shape19*19*6 as represented in Fig. 3. From the 80 potential subclasses in the MS-COCO dataset, only four classes are predicted—"motorcycle," "car," "truck," and "bus"—are used to produce soft labels for each image.

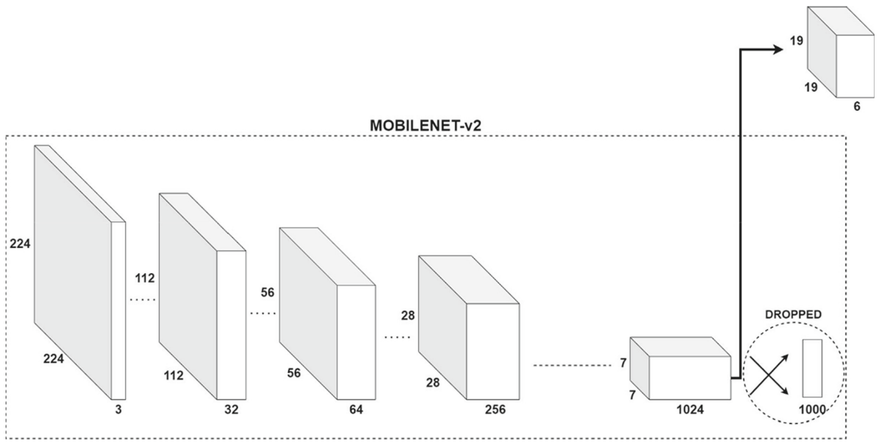

Fig. 3. Yolo architecture. A 19*19*6 tensor replaces MobileNet-v2's last classification

Training

During training, when an object is identified in bounding box, the loss of classification (Eq. 2) in each cell equals the squared variance of the category's conditional probability for every class:

$$L_1 = \sum_{i=0}^{s^2} 1_i^{obj} \sum_{c \in classes} \left(p_i(c) - \hat{p_i}(c)\right)^2 \tag{2}$$

where $1_i^{obj} = 1$ if an object shows up in cell i, otherwise 0. p̂i(c) signifies the conditional likelihood for class c in cell i.

Localization loss (Eq. 3): Localization loss calculates the variances in expected sizes aa well as placements of the boundary boxes.

$$\begin{aligned} L_2 = \lambda_{coord} \sum_{i=0}^{s^2} \sum_{j=0}^{B} 1_{ij}^{obj} \left[\left(x_i - \hat{x_i}\right)^2 + \left(y_i - \hat{y_i}\right)^2 \right] \\ + \lambda_{coord} \sum_{i=0}^{s^2} \sum_{j=0}^{B} 1_{ij}^{obj} \left[\left(\sqrt{w_i} - \sqrt{\hat{w_i}}\right)^2 + \left(\sqrt{h} - \sqrt{\hat{h_i}}\right)^2 \right] \end{aligned} \tag{3}$$

where λ_{coord} in order to compensate up for the loss of boundary box coordinates, increase the weight. $1_{ij}^{obj} = 1$ if the object's identity is determined by the jth border box in cell i, else 0, x, and y represent box offsets. Box dimensions are w and h, respectively.

Confidence loss (Eq. 4): If a box-containing object is found, the following degree of confidence loss occurs:

$$L_3 = \sum_{i=0}^{s^2} \sum_{j=0}^{B} 1_{ij}^{obj} \left(C_i - \hat{C_i}\right)^2 \tag{4}$$

where $1_{ij}^{obj} = 1$ Ci is the border box's rating of the box j in cell i, if the jth border box in cell i is responsible for detecting the object, otherwise 0. The degree of degree of confidence loss is following in the absence of an object being found in the box. (Eq. 5):

$$L_4 = \lambda_{noobj} \sum_{i=0}^{s^2} \sum_{j=0}^{B} 1_{ij}^{noobj} \left(C_i - \hat{C_i}\right)^2 \tag{5}$$

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

When identifying the background, λ_{noobj} weighs down the loss by using the box confidence value of boxes j in cell i, or $\hat{C_i}$.

Equation 1*:* For truth predictions, combined training loss: Lg = L1 + L2 + L3 + L4.

Equation 2*:* Soft label loss:

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

where y_{ijk} denotes the expectation of the student YOLOv3model.

and $\hat{y}_{ijk}$ denotes the expectation of the teacher YOLOv3model.

Equation 3: Combined YOLOv3 training loss:

$$L = L_s * \alpha + (1 - \alpha)L_g \tag{7}$$

where $\alpha \in [0,1]$ is a hyperparameter used to measure soft label loss.

Evaluation Metrics: We computed the AP value on the preprocessed dataset to assess the detection ability of our model. The accuracy of detectors for objects is frequently assessed using the AP, or average precision score.

$$AP = \int p(r)dr \tag{8}$$

3.2 Phase 2 - Vehicle Tracking

The major goal of SORT's pragmatic approach to tracking many objects is to associate them effectively for online as well as real-time applications. Despite merely employing a simple combination of well-known methods for the tracking components, like the Kalman Filter along with Hungarian algorithm, this method achieves an accuracy on par with cutting-edge online trackers. Additionally, relative to other competing methods, it delivers exceptional runtime speed because of its simplicity. The localization and velocity of each vehicle's bounding box were considered by the writers in SORT, and in order to approximate values from previously collected data, they employed Kalman filtering. Then, depending on the Euclidean distance between each of the bounding boxes, they utilized the Hungarian assignment algorithm to assign each one to previously established tracks or, if the distance was too great, to initialize a new track. A recursive method called the Kalman filter determines the current state of a system using noisy measurements. The system's state shows the position and motion of an object in a video stream when it comes to online real-time tracking. The measurements of an object's position, its noise characteristics, and the current state estimation are all inputs to the Kalman filter. The updated state estimate is then generated, considering the fresh measurements, and dampening the impact of measurement noise. The process is described in Fig. 4.

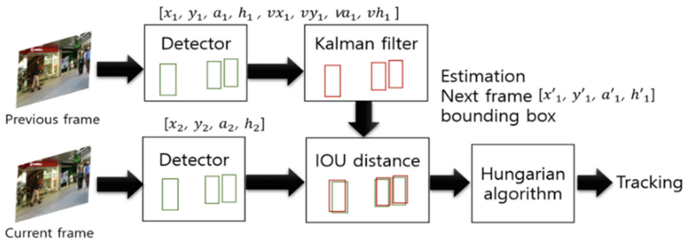

Fig. 4. SORT Tracking

The object detections are assigned to the tracks using Hungarian algorithm. The assignment issue, which is the challenge of assigning a set of items to a set of targets, is solved by a combinatorial optimization algorithm. In the framework of online real-time surveillance, the targets are the previously established tracks, and the objects represent detected in the current frame. Hungarian algorithm places each detection on the line of track that is physically closest to it. The overall distance between each of the detection and the tracks is calculated for this assignment to be as short as possible. The shape 19*19*6 of image which the algorithm allocates detections to tracked objects in a way that reduces the overall distance between them, whereas the Kalman filter offers precise state estimations of the object across time. In computer vision applications like tracking objects in CCTV cameras, autonomous cars, and robotics, this strategy is frequently utilized.

3.3 Phase 3 - Accident Classification

The images of the detected vehicles are extracted using the bounding box obtained from the input frame. After that, each image is categorized based on the extent of damage. An accident was caused by a damaged automobile, according to higher damage status. To categorize identified vehicle photos for damage state, a SVM was trained using a radial basis kernel. The training methodology is covered in detail in the section that follows.

Dataset Used

The classification model was trained using the Accident dataset (Pasheai al., 2019). There are 10,480 photos of both damaged and undamaged autos in the dataset. The majority of the samples in the repository were downloaded from websites. This results, a larger share is made up of excellent photos. The size of the dataset is roughly 1.5 GB. The photos are 640 $\times$ 480 pixels in size and in the JPEG format. Each image in the collection is given a class label (damaged or undamaged) that corresponds to it.

Preprocessing

Every picture in this data set is transformed to grayscale color space and down sampled to 224*224 resolution. Given that the majority of the database's photos were directly.

Most of the content was taken from the internet and was of great quality. We have included nominal noise in Gaussian form to simulate inputs directly acquired from lower-resolution CCTVs to boost the generalization of the model's results in deployment. Such a process has demonstrated a notable improvement in accuracy at the inference time.

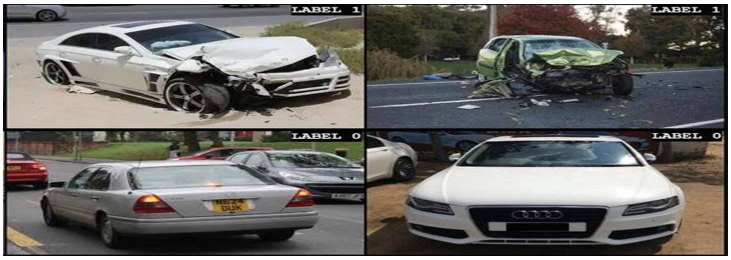

Fig. 5. Label 1 for "damaged vehicle" & Label 2 for "undamaged vehicle"

Training
For the purpose of categorising vehicle photos into damaged and undamaged categories, we trained the SVM using radial basis kernel. The standard configuration in scikit-learn, a Python language tool that is used for training the SVM, is 1.0 for the most important SVM parameter, slackness, and 0.1 for the gamma value of the radial basis function. Figure 5 represents the classification for vehicles.

Evaluation Metrics
To assess the classification performance of the SVM classifier, recall and precision were evaluated. Accuracy and memory are important factors to consider when evaluating models. Precision is with regard to of runtime complexity defined as the group of correctly classified positive instances amongst all occurrences, whereas recall is the percentage of every instance in the positive category that were correctly labelled.

4 Results

YOLOv3 surpasses all other detection methods on the MS-COCO datasets. Therefore, driver is informed runtime complexity as well averaging a startling 30 frames per second on a low-end PC. This results in an average precision (AP) value of 44.8. Simple Online Real Time Tracking, or SORT, is used for the tracking step, and for the stage of classification, various machine learning techniques are compared and shown that the support vector machine technique with a radial basis kernel excels over the opposition with an AUC. Table 2 shows the AP score and runtime of various models when used with ResNetv2 as backbone classifier. Table 3 shows the Precision, Recall and F-score of various classification algorithms.

Table 2. Performance evaluation of several detection algorithms

Model	AP score	Run Time(fps)
Faster R-CNN	34.7	15
Single Shot Detectors	31.2	19
Yolo v3	44.8	30

Table 3. The accident classification model's performance

Model	Precision	Recall	F1-score
Random Forest	0.85	0.79	0.82
Decision Tree	0.70	0.69	0.70
SVM(linear) Classifier	0.82	0.81	0.82

4.1 AUC-ROC Analysis

A typical statistic for assessing the efficacy of binary classification algorithms is the area under a curve (AUC). It evaluates the model's ability to distinguish between positive and negative categories. AUC has a range of 0 to 1, with 0.5 suggesting that the algorithm is no better than guessing at random and 1 suggesting perfect classification. Here we are classifying between two labels accident vs non-accidents. Figure 6 represents the AUC using various algorithms - decision tree attains 0.91, random forest attains 0.70, linear SVM achieves 0.91 and SVM achieves 0.92.

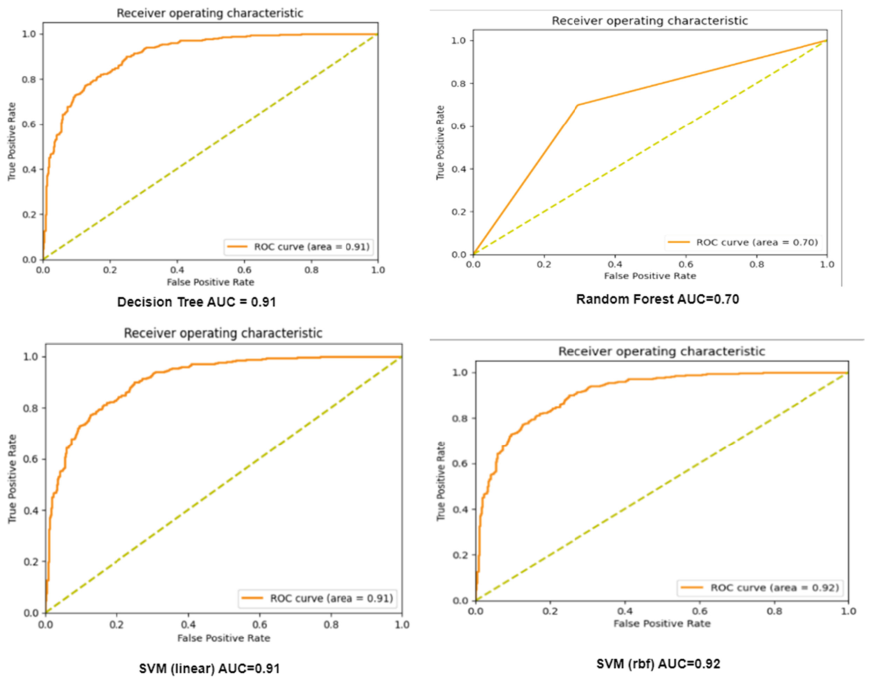

Fig. 6. Illustration of AUC using various algorithms

5 Conclusion

The relationship between computational overhead and detection accuracy affects nearly every AAD system. The study offers a way for creating a real-time, AAD system that is both reliable and affordable in terms of computing. The suggested system seeks to address the trade-off between computing overhead and detection accuracy by segmenting the model into three phases – Detection phase, tracking phase, and classification phase) and employing unique algorithm for each stage. YOLO v3 for the Detection stage are introduced. SORT is used for the tracking stage. The research evaluates various machine learning methods for the classification stage and determines that the SVM, which employs a radial basis kernel, is the best performance. Overall, the proposed method offers a viable solution to the problems of processing overhead and detection precision in AAD systems. The suggested method strikes a balance between computational effectiveness and accuracy, opening the door for the creation of useful and affordable real-time accident detection technologies. Future research directions in this field may include exploring the integration of other data sources, such as LiDAR and radar, to enhance the system's performance and robustness. Additionally, the development of more efficient and lightweight deep learning models, such as those based on neural architecture search, can help address the computational challenges associated with real-time processing. Finally, the incorporation of reinforcement learning and other advanced machine learning techniques can enable the system to adapt and improve its performance over time, further enhancing its effectiveness in detecting and preventing accidents.

As we continue to explore the possibilities of vision-based accident detection systems using deep learning, it is important to consider their broader implications and applications. These systems can be integrated with intelligent transportation systems (ITS), such as traffic signal control and dynamic traffic routing, leading to more efficient traffic management and improved road safety. Real-time accident detection enables faster emergency response, potentially saving lives and reducing the severity of injuries. The data collected by these systems is valuable for insurance companies and law enforcement agencies, aiding in determining accident causes and liability assessment. With the rise of autonomous vehicles, vision-based accident detection systems can ensure their safe operation by proactively detecting and avoiding potential accidents. Additionally, the data gathered can provide insights for urban planning and infrastructure development, helping design safer and more efficient transportation networks.

References

1. Road traffic injuries. World Health Organization, 21 June 2021. https://www.who.int/news-room/fact-sheets/detail/road-traffic-injuries. Accessed 20 June 2023
2. Caroselli, Beachler & Coleman, L.L.C., 22 September 2021. https://www.cbmclaw.com/what-percentage-of-car-accidents-are-caused-by-human-error. Accessed 01 July 2023
3. Kapri, P., Patane, S., Shalom, A.: Accident Detection & Alert System, IEEE (2018)
4. Fernandes, B., Gomes, V., Ferreira, J., Oliveira, A.: Mobile Application for Automatic Accident Detection and Multimodal Alert, IEEE (2015)

5. Fogue, M., Garrido, P., Martinez, F.J.: System for Automatic Notification and Severity Estimation of Automotive Accidents, IEEE (2014)
6. Sankpal, P.M., More, P.P.: Accident avoidance system using IR transmitter. IJRASET **5**(IV) (2017)
7. Bewley, A., Ge, Z., Ott, L., Ramos, F., Upcroft, B.: Simple online and realtime tracking. In: 2016 IEEE International Conference on Image Processing (ICIP), pp. 3464–3468 (2016). https://doi.org/10.1109/ICIP.2016.7533003
8. Dogru, N., Subasi, A.: Traffic accident detection using random forest classifier. In: 2018 15th Learning and Technology Conference (L&T), pp. 40–45 (2018). https://doi.org/10.1109/LT.2018.8368509
9. Chand, D., Gupta, S., Kavati, I.: Computer vision based accident detection for autonomous vehicles. In: 2020 IEEE 17th India Council International Conference (INDICON), pp. 1–6 (2020). https://doi.org/10.1109/INDICON49873.2020.9342226
10. Tsuji, T., Hattori, H., Watanabe, M., Nagaoka, N.: Development of night-vision system. IEEE Trans. Intell. Transp. Syst. **3**(3), 203–209 (2002)
11. Wakim, C., Capperon, S., Oksman, J.: A Markovian model of pedestrian behavior. In: Proceedings of the IEEE International Conference on Systems, Man, and Cybernetics, pp. 4028–4033, October 2004
12. Antonini, G., Venegas, S., Thiran, J.P., Bierlaire, M.: A discrete choice pedestrian behavior model for pedestrian detection in visual tracking systems. In: Proceedings of the Advanced Concepts for Intelligent Vision Systems, Brussels, Belgium, September 2004
13. Shimizu, H., Poggio, T.: Direction Estimation of Pedestrian from Images, AI Memo 2003-020, Massachusetts Institute of Technology, August 2003. http://cbcl.mit.edu/cbcl/publications/aipublications/2003/AIM-2003-020.pdf
14. Hui, Z., Yaohua, X., Lu, M., Jiansheng, F.: Vision-based real-time traffic accident detection. In: Intelligent Control and Automation (WCICA), 2014 11th World Congress on, pp. 1035–1038, June 2014
15. Ren, J., Chen, Y., Xin, L., Shi, J., Li, B., Liu, Y.: Detecting and positioning of traffic incidents via video-based analysis of traffic states in a road segment. IET Intell. Transp. Syst. **10**(6), 428–437 (2016)
16. Shafiee, M.J., Chywl, B., Li, F., Wong, A.: Fast YOLO: a fast you only look once system for real-time embedded object detection in video. J. Comput. Vis. Imaging Syst. **3** (2017). https://doi.org/10.15353/vsnl.v3i1.171
17. Lin, C.-J., Jeng, S.-Y., Lioa, H.-W.: A real-time vehicle counting, speed estimation, and classification system based on virtual detection Zone and YOLO. Math. Probl. Eng. **2021**, 1–10 (2021). https://doi.org/10.1155/2021/1577614
18. Pillai, M.S., Chaudhary, G., Khari, M., et al.: Real-time image enhancement for an automatic automobile accident detection through CCTV using deep learning. Soft. Comput. **25**, 11929–11940 (2021). https://doi.org/10.1007/s00500-021-05576-w
19. Singh, P.K., Gupta, A., Upadhyay, M., Jain, A., Khari, M., Lamba, P.S.: Multimodal driver drowsiness detection from video frames. J. Mob. Multimed. **19**(02), 567–586 (2022). https://doi.org/10.13052/jmm1550-4646.19210
20. Behrendt, K.: Boxy vehicle detection in large images. In: 2019 IEEE/CVF International Conference on Computer Vision Workshop (ICCVW), pp. 840–846 (2019)