An IoT Based Vehicle Accident Detection and Classification System using Sensor Fusion

Nikhil Kumar, Debopam Acharya, and Divya Lohani

Abstract—**Road accidents are a leading cause of death and disability among youth. Contemporary research on accident detection systems is focused on either decreasing the reporting time or improving the accuracy of accident detection. Internet of Things (IoT) platforms have been utilized considerably in recent times to reduce the time required for rescue after an accident. This work presents an IoT-based automotive accident detection and classification (ADC) system, which uses the fusion of smartphone's built-in and connected sensors not only to detect but also to report the type of accident. This novel technique improves the rescue efficacy of various emergency services such as EMS (emergency medical services), fire stations, towing services, etc., as knowledge about the type of accident is extremely valuable in planning and executing rescue and relief operations. The emergency assistance providers can better equip themselves according to the situation after making an inference about the injuries sustained by the victims and the damage to the vehicle. In this work, three machine learning models based on Naïve Bayes (NB), Gaussian Mixture Model (GMM) and Decision Tree (DT) techniques are compared to identify the best ADC model. Five physical parameters related to vehicle movement i.e. speed, absolute linear acceleration (ALA), change-in-altitude, pitch, and roll, have been used to train and test each candidate ADC model to identify the correct class of accident among collision, rollover, fall-off, and no-accident. NB-based ADC model is found to be highly accurate with 0.95 mean F1-score.**

Index Terms—**Accident detection and classification, sensor fusion, Naïve Bayes, GMM, Decision Tree, Internet of Things**

I. INTRODUCTION

DEATHS and disabilities by road accidents are increasing with each passing year. An increase in population and per capita incomes has led to an increase in ownership and the presence of vehicles on roads. Greater traffic volumes, over-speeding, reckless and drunken driving, driver fatigue, poor road infrastructure and the presence of animals on roads are some reasons that are responsible for road fatalities. According to the World Health Organization (WHO), the percentage of road accident fatalities to the total number of deaths worldwide has increased by 2.2% [1]. Approximately 1.35 million people die due to road accidents every year [2].

In many cases, human lives are lost in road accidents due to delays in emergency medical assistance. According to Golden Hour Principle [3], there is a high probability that timely medical and surgical aid can avoid death during the golden hour, which is the period after the traumatic injury. A decrease in the response time of emergency medical care can reduce the probability of death by one-third on an average [4]. The percentage of people who die before reaching the hospital in low- and middle-income countries is more than twice as compared to high-income countries [2]. In the recent past, information and communication technology such as IoT has been used to decrease the accident rescue time. IoT is an interconnection of vast variety of embedded and smart devices such as computers, smartphones, smart sensors and actuators, embedded processors, etc. with the modern internet. IoT is a potential medium for tracking and control of smart automobiles that can link any connected physical unit to a control server [5]. Most researchers have confined their work to improving the accuracy of accident detection, estimating the severity of road accidents or minimizing the rescue time post-occurrence of an accident [6], [7]. In addition, most systems to detect and report road accidents are expensive and limited to high-end vehicles. Another drawback of the current systems is their inability in identifying the type of accident as a collision, rollover or a fall-off event. Merely reporting the occurrence of an accident event severely limits the ability of the emergency rescue workers to provide the victims with the right kind of rescue support and medical aid. The information about the type of accident sustained provides valuable information about the damages sustained by the vehicle and the extent of injuries sustained by the victims.

This research work answers the following questions:

1. Can there be an inexpensive accident detection and classification system that can be retrofitted to any vehicle?
2. Which is the best-suited machine-learning classification model that can accurately detect and classify the road accident type?
3. How to automatically report the occurrence of road accident with its type and location to the relevant agencies by sending the emergency notification if the accident victims are incapacitated?

Contribution of this work: Research work on developing the mechanisms for prediction, prevention, detection and management of road accidents are predominantly focused on either enhancing precision or reducing rescue time following the occurrence of road accidents. In this study, we have proposed a smartphone-based end-to-end IoT system architecture to detect and notify automotive accidents. The work also focuses on identifying the best machine-learning ADC model based on either of three different classifiers, namely GMM, DT, and NB. This classification can be extremely helpful in the planning and execution of rescue operations. The proposed solution is inexpensive and efficient

N. Kumar and D. Lohani are with Department of Computer Science and Engineering, School of Engineering, Shiv Nadar University, G. B. Nagar, Uttar Pradesh, India (e-mail: nk438@snu.edu.in; divya.lohani@ieee.org)

D. Acharya is with School of Computing, DIT University, Dehradun, Uttarakhand, India (email: dr.debopamacharya@dituniversity.edu.in)

that can be easily retrofitted in any class and model of the vehicle. Experiments and analysis of the results have shown that in all proposed solutions NB-based model is highly accurate in detection and classification of road accidents.

This paper is organized as follows. Major related work in the area of automatic vehicle accident detection and multi-sensor fusion is described in Section II. The architecture of the proposed IoT system, workflow diagram of the system, model variables and pre-processing methods are explained in Section III. The hardware and software setup of the system is proposed in Section IV. Next, Section V describes the possible accident scenarios, and then there is a description of the experimental setup and data collection in Section VI. Three different accident detection and classification models, based on GMM, DT and NB classifiers, are proposed in Section VII. Results, comparative analysis of proposed models, and discussion are presented in Section VIII while conclusion and future work are laid out in Section IX.

II. Related Work

Research work related to the detection, localization, reporting, modeling, and analysis of road accidents is discussed in this section. Several authors have reported Smartphone-based accident detection systems. F. Aloul et al. [8] developed an Android application that uses the accelerometer data to develop an accident detection model based on Dynamic Time Warping (DTW) and Hidden Markov Models (HMM). An SMS, which includes accident location and severity, is sent to the EMS, police and family on detection of an accident. F. Bhatti et al. [9] have used built-in speed, location, pressure, sound, and g-force sensors of a smartphone to develop a low cost, portable solution that detects and reports an accident to the nearest hospital.

A. Shaik et al. [10] proposed an IoT-based vehicle collision detection system that uses a ADXL345 accelerometer to detect the collision and GPS to detect and send the accident location to the nearest ambulance via internet. S. Sharma and S. Sebastian [11] proposed an algorithm for vehicle accident detection that uses ADXL345 accelerometer and vibration sensor to detect the vehicle crash, and heart rate sensor to monitor health of the passenger, and send an emergency SMS or voice call to the family. An accelerometer and an ultrasonic sensor are used by E.K. Priya et al. [12] in an AtMega162-microcontroller-based accident detection system to notify the nearest first aid center about the accident location via SMS.

To reduce delay in reporting a road accident, B.K. Dar et al. [13] have proposed a fog computing-based approach to build a low-cost smartphone system. The system uses an Android application that takes inputs from smartphone sensors to detect an accident. The system locates and notifies a nearby hospital using GPS when an accident is detected. B. Fernandes et al. [14] have developed an Android application, which receives inputs from the smartphone sensors to detect collisions and rollovers, as well as receive road hazard warnings issued by vehicles in the vicinity.

When a good Internet connection is limited to some nearby vehicles, inter-vehicle communication approaches, such as VANET (Vehicular Ad-hoc Network) and IoV (Internet of Vehicles), may help vehicles carry out reporting of accidents via each other. P. Fabian et al. [15] have introduced a programmable objective function (POF) and developed an architecture inspired from a software-defined network (SDN) approach to improve the quality of service (QoS) in dynamic IoV networks, which can be very important while vehicle reporting the vehicle accidents to each other. A. Rachedi and H. Badis [16] have proposed a programmable and hybrid architecture based on the virtual backbone (VB) formed by selected parked vehicles. Architecture is used to reduce the use of cellular communication link in IoV, which is based on Connected Dominated Sets (CDS) and Software-Defined Vehicular Networks (SDVN) approaches. T. Mekki et al. [17] have discussed a vehicular cloud access problem that is modeled as an evolutionary game, and proposed two vehicular cloud access (VCA) algorithms based on evolutionary game (EG-VCA), and distributed Q-learning (QL-VCA).

D. Acharya et al. [18] and H.A. Ibrahim et al. [19] have developed systems that use accelerometer and gyroscope data to detect and report vehicle rollover events. J. Smolka et al. [20] present a smartphone-based collision detection system using accelerometer, magnetometer and GPS module to reduce the number of false alarms. S. Sadek et al. [21] have used real-time video footage of traffic surveillance systems to construct histogram of flow gradient (HFG). The HFG features are used as inputs to develop the statistical logistic regression model to predict road accidents. None of these systems reports the ability to be able to determine the type of accident, which can be useful in planning the rescue type and estimating the number of emergency services required.

The fusion of sensor data for detection and analysis of road accidents is not reported in any major work, which can help to improve the ADC performance. F. Felisberto et al. [22] present a system, which uses the fusion of data from a network of wireless sensors to detect accidents in elderly people. The authors claim that the proposed solution is nearly 100% accurate in detecting normal falls. I.G. Damousis et al. [23] have used a fuzzy expert system for the fusion of the most significant features extracted from the eyelid activities of vehicle drivers. The system is found to be efficient in predicting sleep onset and accidents. Z. Zhang et al. [24] have attempted the fusion of multi-source data by combining traffic metrics and social media tweets for real-time detection of road accidents. Using support vector machines (SVM) as the classification model and implementing 5-fold cross-validation, the authors claim an increase in prediction accuracy by the integration of the two data sources.

S. Moulik et al. [25] have developed an IoT system that uses the fusion of multiple ultrasonic sensors. The system uses infrared transmitter-receiver pairs and a fuzzy inference system to detect the accidental fall of humans. The multi-sensor fusion is claimed to achieve an improvement of 16% as compared with existing approaches.

Although, previous works have proposed some systems for collision/rollover detection and notification, to the best of our knowledge there is no system which classifies these accidents as collision, fall-off, rollover, and no-accident. This work presents a novel smartphone sensor fusion based solution for detection and categorization of accident.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2020.3008896, IEEE Internet of Things Journal

III. Architecture of the System

A. Architecture of our proposed IoT System

An IoT architecture is proposed in Fig. 1 to address the problem of vehicle accident classification. We have used a modern smartphone and a prefabricated sensor platform termed as Sensordrone [26] to obtain the values of different physical parameters relevant to vehicle motion. The Android smartphone contains several integrated sensors, such as accelerometer, gyroscope, magnetometer, GPS, etc., which can be used to determine the speed, direction, rotation, and g-force, etc. of the vehicle. The Sensordrone remains attached to the smartphone via Bluetooth connection to send the data to the smartphone. Most of the processing is handled by the smartphone. Processing in smartphone substantially decreases the internet resource (e.g., bandwidth, etc.) usage by transmitting only relevant details such as venue, name, nature of the accident, etc., to the IoT server. The IoT server delivers emergency alerts to various emergency services such as EMS, local police station and fire service, and other receivers such as relatives, insurers, blood donor, towing service, etc., after assessing the situation.

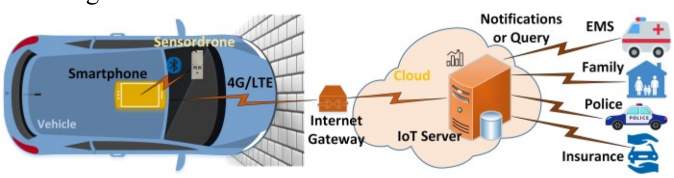

Fig. 1. Architecture of proposed IoT system.

B. Operating Workflow Diagram

The operating flow diagram of the proposed ADC system is depicted in Fig. 2. The feature vector of our classification model includes five parameters, namely speed, ALA, change-in-altitude, pitch, and roll, to detect and classify a vehicle accident. The system is trained and a knowledge base having four accident classes is created by providing the training-dataset to the machine learning classifier. After training, the system model is tested against the testing dataset; and then accuracy is measured to ascertain the delivery of valid notification to the emergency services, in case of an accident.

C. Model Variables

1) Speed

It is quite logical that when a car experiences a serious accident, whether it is a collision, a rollover or a fall-off, its speed would become zero ultimately. Smartphone's GPS has been used to measure the speed of the vehicle. Each GPS device receives NMEA (National Marine Electronics Association) sentence from satellites containing information related to the location, velocity and time of the device [27]. Each class of GPS devices does have its own NMEA sentence. The most common NMEA sentence that is used by most of the Android devices is $GPRMC. Following is an example of the $GPRMC NMEA sentence, where the velocity is 018.8 knots:

```
$GPRMC,152844,A,1831.335,N,07734.331,E,
018.8,054.4,030220,003.1,W*6A
```

2) Absolute Linear Acceleration (ALA)

Whenever a running vehicle drop from a height, the orientation of the vehicle will not be the same as it was on the road before falling. In such a case, the vehicle cannot remain parallel to the gravitational axes (X, Y, and Z) and deceleration due to the impact that can be distributed in more than one axis. Therefore, it is very difficult to assess the precise value of the deceleration when it is distributed in X, Y, and Z components. To address this problem, ALA (or Signal Magnitude Vector [28]) is calculated with the help of X, Y, and Z components of deceleration. ALA is independent of the vehicle's orientation during fall-off or collision directions (whether the vehicle is moving or steady). ALA shows the acceleration characteristic parallel to the direction of impact. This absolute quantity is a resultant vector of X, Y and Z components of deceleration. ALA is determined as:

$$ALA = \sqrt{\left(DEC_X\right)^2 + \left(DEC_Y\right)^2 + \left(DEC_Z\right)^2} \tag{1}$$

where DEC_X, DEC_Y, and DEC_Z are the decelerations at X-axis, Y-axis, and Z-axis respectively. Smartphone's accelerometer has been used to measure the ALA whose measurement unit is g ($\approx$9.80665 m/s^2). It is mentioned in the literature that when a vehicle collided with an object (or any solid surface) at the speed of 23 km/h or more, the intensity of deceleration always cross 5g [29].

3) Change in Altitude in One Second

Knowledge of altitude-shift enables the system to predict the fall-off of the vehicle. During the fall-off, the altitude of the vehicle is the most accountable feature that shifts drastically. Although GPS can also be used to determine the altitude, GPS is not acceptable when the position stays the same while the altitude shifts. No specialized instrument is present that can determine altitude solely, but it can be determined using atmospheric pressure P and temperature. In

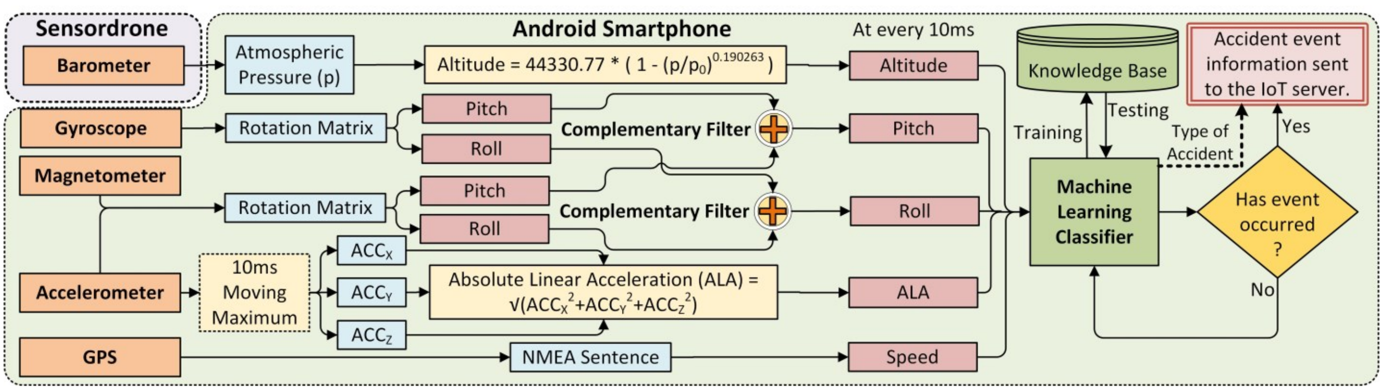

Fig. 2. Workflow diagram of accident classification system

this experiment, the altitude is calculated by using the pressure *P* measured by the Sensordrone's barometric altimeter. The atmospheric temperature is neglected because it does not substantially change at same coordinates. If P_0 is reference pressure recorded at the sea level, final formula of calculating the altitude would come out as provided by H. Ye et al. [30].

$$Altitude = 44330.77 * \left(1 - \left(\frac{P}{P_0}\right)^{0.190263}\right) \quad (2).$$

4) Roll, and Pitch

The rotation angles of an object around the X- and Y-axis are known as roll and pitch, respectively. Roll and pitch are calculated by the fusion of rotation information provided by the gyroscope, accelerometer, and magnetometer using a complementary filter. The rotation matrix rotates the Euclidean space counterclockwise around the center of the Cartesian coordinate system by angle θ. We found that if either roll or pitch is greater than 90°, a rollover happened. The longitudinal axis is always supposed to be X-axis and the lateral axis is always supposed to be Y-axis (Fig. 3(left)).

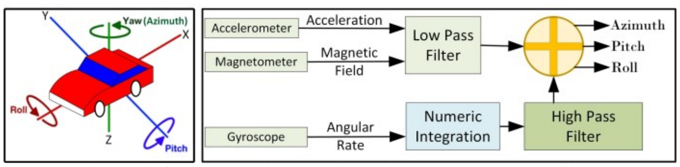

Fig. 3. (left) yaw, pitch and roll; (right) complementary filter.

D. Pre-processing Methods

1) Sensor Fusion using Complementary Filter

The direction and orientation of the vehicle can be determined by pitch, roll, and azimuth (yaw). The longitudinal, lateral and vertical axes of the vehicle's body are expressed by X, Y, and Z respectively. Roll, pitch and azimuth (yaw) are the rotating angles around the X-, Y-, and Z-axis respectively (Fig. 3 (left)). Three inertial sensors, viz., gyroscope, accelerometer, magnetometer have been used to determine the orientation of the vehicle.

Despite their respective properties, owing to their shortcomings and drift in measurements, no stand-alone inertial sensor is adequate to calculate the orientation effectively. The magnetometer can measure azimuth precisely, but not pitch and roll, whereas accelerometer can calculate pitch and roll precisely, but not azimuth. Collectively, all sensors can calculate rotational speed and rotational angle, about all three axes. At high frequency, short-term estimates of the combination of the accelerometer and magnetometer are not accurate owing to a few small external factors, except the g-force. A low-pass filter (only makes low-frequency signals from 0 Hz to its threshold level) is required for correction because the combination of both of these inertial sensors works well at low frequency.

Another sensor that can calculate the angular momentum about the X-, Y- and Z-axis is the gyroscope, but it is also exposed to strong drift. Gyroscope operates well at a high rate, so high-pass filtering (allows only high-frequency signals from its threshold frequency) is required to fix the results because the measurements start drifting in long run.

Each sensor has its pros and cons, but their fusion can improve the performance anyway [31]. In this research, a complementary filter [32] is used, consisting of both a low-pass filter and a high-pass filter (Fig. 3 (right)). The complementary filter provides data that are more precise by applying the weight fractions to the outputs of low-pass and high-pass filters. If A_{gyro}, P_{gyro}, and R_{gyro} are the corresponding azimuth, pitch, and roll provided by the high-pass filter, and A_{acc_mag}, P_{acc_mag}, and R_{acc_mag} are the corresponding azimuth, pitch, and roll provided by the low-pass filter, then the resultant azimuth, pitch, and roll determined by the complementary filter are as follows:

$$Azimuth = \alpha * A_{gyro} + (1-\alpha) * A_{acc\_mag} \quad (3),$$

$$Pitch = \alpha * P_{gyro} + (1-\alpha) * P_{acc\_mag} \quad (4), \text{ and}$$

$$Roll = \alpha * R_{gyro} + (1-\alpha) * R_{acc\_mag} \quad (5).$$

The gyroscope gives more stable value than accelerometer and magnetometer collectively, and weightage of gyroscope must be near 0.95 while using complementary filter [33]. We have tried to calibrate the combination for different values of α and find that this combination was showing smoother curve of pitch, roll and azimuth at $\alpha = 0.98$.

2) Maximum Deceleration using Ten Millisecond Moving Maximum

The standard signal generation rate of an accelerometer might be more than 2000 Hz. It is quite challenging to process the data and report the maximum ALA value at such a high frequency, especially when we need accurate decisions and our device has limited resources and processing capabilities. M. Iyoda et al. [34] have used a 10-millisecond moving average technique to deal with this problem; however, the moving average can diminish the ALA's peak value. To attain the highest possible ALA, we are using a 10-ms Moving Maximum method that registers the peak value for each and every 10 ms time window as ..., *S-2, S-1, S, S+1*, ... (Fig. 4).

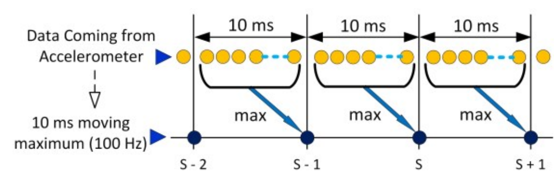

Fig. 4. Ten millisecond moving maximum.

IV. Hardware and Software Setup of the System

A. Hardware Setup

In this work, a complete hardware setup comprises a *SAMSUNG Galaxy S8* Android smartphone, a *Sensordrone* and a 1:12 scaled RC car. The hardware setup is depicted in the inset figure of Fig. 6.

1) SAMSUNG Galaxy S8 Android Smartphone

In this work, besides of external sensors, the *SAMSUNG Galaxy S8* Android smartphone's inbuilt 9-axis inertial sensor and GPS sensor have been used. Inertial sensor, whose measuring range is ±179° and ±16g, is used to measure the pitch, roll, and deceleration, and a GPS sensor is used to determine the vehicle's speed and location. 1.9 GHz clock speed of *Exynos 8895* microprocessor and 4GB of RAM makes it a suitable computing device for our experiments.

2) *Sensordrone*

Sensordrone is a tiny sensor hub that includes seven separate in-house sensors to assess twelve different environmental variables, such as temperature, humidity, CO, atmospheric pressure, light, proximity, etc. [26]. *Sensordrone* is a programmable device, which has a UART port to attach an additional sensor (e.g. CO_2 sensor) and Bluetooth to establish a connection with any processing device. In this research, only barometric altimeter sensor (measurement range: 26kPa to 126kPa) of *Sensordrone* is used to determine the altitude of the vehicle.

3) *RC Car*

A 1:12 scaled toy RC (Radio Controlled) Car [35] is used to mimic real-life accident scenarios since conducting the accident experiments with an actual car is neither viable nor budget-effective. This RC car is made of heavy-performance ABS material and has metallic shock absorbers. Configuration of RC car is specified in Table I.

TABLE I. RC CAR SPECIFICATIONS

Vehicle Parameter	Value
Brand Name	GPTOYS Foxx S911 RC Car
Dimension (L * W * H)	31.0 cm * 26.5 cm* 15.0 cm
Weight	1.078 Kg
Scale Ratio	1:12
Track Width	22.0 cm
Wheelbase	20.7 cm
Ground Clearance	40.0 cm
Maximum Speed	53 km/h
Transmitter Frequency	2.4 GHz 4Ch
Operating Distance	80 m
Maximum Turning Angle	45 degrees

B. *Software Setup*

Two Android applications, *SNUSense*, and *SNUAlertApp* have been developed for sending and receiving emergency notifications. SNUSense is required to be installed on the accident victim's smartphone to collect and process the sensor's data. All the pre-processing methods such as ALA calculation, complementary filter (sensor fusion), moving-maximum function, and machine learning classifiers (feature fusion) are programmed in the *SNUSense* application. *SNUAlertApp* is required to be installed on accident rescuers' Android smartphone so that timely rescue operation can be initiated after locating the place.

1) *SNUSense Android Application*

An Android application, namely *SNUSense*, has been developed for capturing and analyzing various parameters such as rotation angle (roll, pitch, and azimuth), speed of the vehicle, noise, deceleration at different axes, atmospheric pressure, altitude, temperature, latitude, longitude, humidity, CO_2, CO, light intensity, etc. by utilizing Android smartphone sensors and *Sensordrone* (Fig. 5 (left)). It continuously processes the streamed data coming from the Sensordrone and smartphone sensors and makes an estimate using the vehicle's five characteristics, that is, speed, ALA, change-in-altitude, pitch, and roll. After a positive detection of the accident event, SNUSense sends the name of the victim, type and location of the vehicle accident to the SNUSense IoT server (*Google Firebase* [36]) using 4G internet connection. We can see in the user dashboard that in case of false alarm, victim can abort the notification by pressing the STOP button (Fig. 5 (center)). IoT server further sends the notification (using *Firebase Cloud Messaging* service) to the intended recipients such as EMS, police, family, etc. after analyzing the data (such as the severity of the accident based on the force of impact and the speed). It also collects all sensors' data in a CSV (comma-separated values) file in the smartphone's storage.

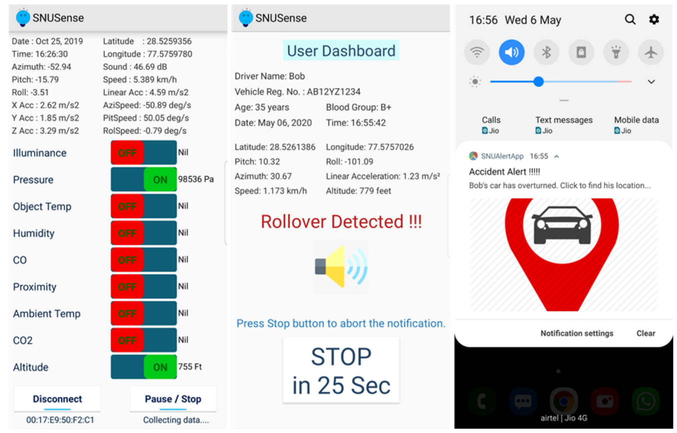

Fig. 5. (left) Data collection; (center) user dashboard; (right) notification.

2) *SNUAlertApp Android Application*

Another Android application, *SNUAlertApp*, has been created to receive accident alerts as shown in Fig. 5(right). The alert notification contains the coordinates of the accident, the name of the driver, and the type and severity of the accident. A simple touch on the notification launches the *Google Maps* with a tracker icon on it so that rescuers can conveniently locate the place of accident.

V. ACCIDENT SCENARIOS

Depending on the five variables described in sub-section III-C, this research focuses only on the identification and classification of four different vehicle accident situations: collision, fall-off, rollover, and no-accident. Table II describes the actions and criteria of five factors for each accident situation to be classified.

Thresholds of speed and roll/pitch are identified by performing accident tests multiple times with our RC car setup. If any fatal accident occurs, ideally speed of vehicle becomes zero instantly. Threshold for speed is considered 2km/h because GPS readings does not stabilize immediately and slightly fluctuates around for a while. In the literature, for rollover, different researchers have considered different angles of roll / pitch to define the rollover. In this work, we are considering the rollover for angles greater than or equal to 90 degrees, as we are considering not partial but complete rollover events. For collision, it is mentioned in the literature that if a vehicle collides with a solid obstacle with a speed of more than 23km/h (i.e. 21feet/sec) then it will generate a deceleration of more than 5g and the collision is said to be severe [29]. From the following equation of velocity, we can observe that if a vehicle falls from 8 feet with an initial speed of 0, it achieves the speed of more than 21feet/sec in almost 0.8 sec, which is minimum speed to generate 5g deceleration.

$$velocity_{final} = velocity_{initial} + g * t \quad (6).$$

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2020.3008896, IEEE Internet of Things Journal

So, threshold of parameter, i.e. *change-in-altitude,* is considered a minimum 8ft. Although there may be other sub-classification of the accidents, this study is restricted to collision, fall-off, and rollover events. During the implementation of the classification, the lag between the reading timings of different parameters is well-thought-out.

TABLE II. ACCIDENT DETECTION AND CLASSIFICATION CRITERIA

Parameter (Threshold)	ALA (5 g)	Speed (2 km/h)	Pitch / Roll (90°)	Altitude Change in 1 sec (8 feet)
Collision	> 5 g	< 2 km/h	anything	< 8 feet
Fall-off	> 5 g	< 2 km/h	anything	> 8 feet
Rollover	Anything	< 2 km/h	>= 90°	anything
No accident	Combinations other than above mentioned			

VI. EXPERIMENTAL SETUP AND DATA COLLECTION

A. Experimental Setup

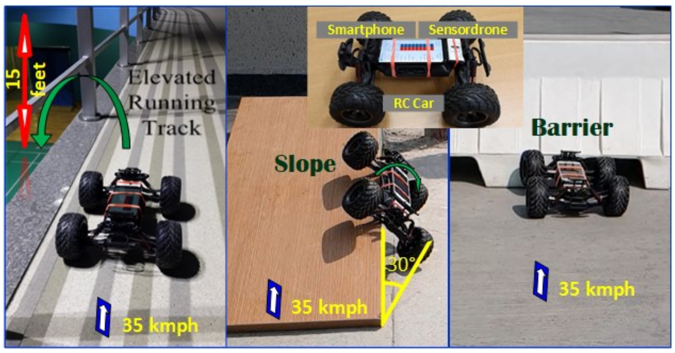

Fig. 6. (left) fall-off setup; (center) rollover setup; (right) collision setup.

To accurately detect collision and fall-off, the smartphone is placed near the center of gravity (CG) of the vehicle (in our case, on the armrest between the front seats) as both collision and fall-off events leads to deceleration, and it is measured with the help of smartphone's built-in accelerometer sensor. Although the direction of impact on the vehicle cannot be foreseen, the deceleration measurement would be more accurate if the direction of impact is same as the direction of the accelerometer sensor installation. Similarly, if the impact is on the opposite side of the sensor position, due to the size and build of the vehicle, the deceleration output may be less than the actual. The National Highway Traffic Safety Administration (NHTSA), the nodal agency of the US Department of Transportation, recommends that inertial sensors be placed near the CG [37], for all crash tests. As shown in Fig. 6, we have also followed this recommendation for our experimental RC car setup and tied up the smartphone on chassis of the RC car. For rest of the sensors used in our experiment, their placement within a vehicle does not have a significant impact on the accuracy of their data values.

All the tests were performed at the speed of 35.1 km/h because in the literature, minimum threshold speed to detect collision (or airbag deployment) of vehicle is ≈23 km/h [29]. It is considered that speed below 23 km/h will not be enough to create a severe collision impact. As speed will increase, severity of accident will increase proportionally. Because this research does not focus on severity monitoring, considering the speed similar to the minimum speed would be sufficient to perform the accident tests. Another reason of considering experimental speed 35.1 km/h is the RC car's limitations. Although RC car's maximum speed is 53 km/h (which cannot be set to a specific speed), but due to the weights of smartphone and Sensordrone, it comes down to 35.1 km/h.

B. Data Collection

To collect the head-on collision data, we have used a plastic road barricade as a barrier (Fig. 6(right)), and the RC car setup is crashed into it. Fig. 7 is showing the change in ALA, speed, and change-in-altitude during collision event. After the collision, a sudden decrease in the speed and increase in the ALA is clearly noticeable in the graph. The drawn red line reflects the moment when the collision occurred.

The fall-off experiment is conducted on the elevated running track of the Indoor Sports Complex (ISC) at Shiv Nadar University, India (Fig. 6(left)). The RC car setup was dropped onto the wooden tennis court from the running track elevated by 15-foot height by diverting it to the left side with a 45-degree angle on the track. Fig. 8 reflects the fall-off case, with changes in ALA, speed, and altitude. The gradual decrease in speed is showing the duration of fall, 15 feet change in altitude (the difference between the altitude before the fall and after the fall) and sudden increase in ALA is showing the percussion on the tennis court after the fall. The dotted red line indicates the moment of the percussion. One second sliding window has been used to record the change in altitude because in 1sec the falling object can attain the final speed of more than 30 km/h, which is far enough to create the impact force of more than 5*g*.

To conduct the rollover experiments, we have used a 4 cm thick wooden board that was inclined 30° with the flat ground surface (Fig. 6(center)). Fig. 9 demonstrates changes in speed, roll, and pitch angles during the rollover event. The sudden increase in the roll angle is the perfect indicator of rollover incident. The red dotted line in the graph shows the instant of the rollover activity.

Every sort of accident experiment is repeated 30 times to collect the data and 1167 observations has been recorded. Each plot in Fig. 7, 8 and 9 are plotted using raw data samples received from the sensors, which are showing multivariate time series to observe the trends of different model parameters during different type of accidents.

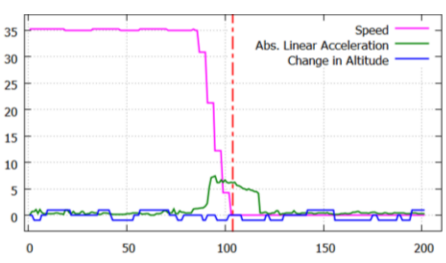

Fig. 7. Collision.

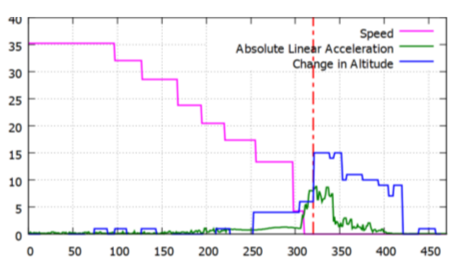

Fig. 8. Fall-off.

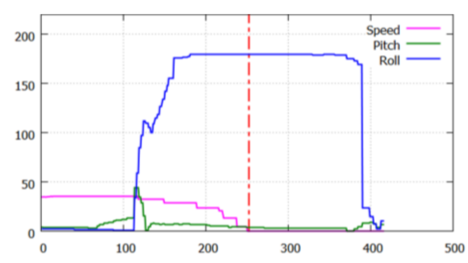

Fig. 9. Rollover.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2020.3008896, IEEE Internet of Things Journal

VII. Accident Detection and Classification Models

In this work, three different classification models, namely GMM (a clustering model), DT (a logic-based model), and NB (a probabilistic model) has been compared to find best learning model to address the vehicle's accident detection and classification (ADC) problem. Five parameters of training data-set, that is, change-in-altitude, pitch, roll, speed, and ALA has been used to train every classification model. Separate testing dataset of each accident event has been used to assess the accident detection and classification performance of the models. 90% observations are used for training the models and 10% observations are used for testing the models.

A. Gaussian Mixture Model (GMM) based ADC Model

GMMs are among the most statistically matured methods for clustering and density estimation [38], [39]. They model the probability density function (PDF) of observed data points using a multivariate Gaussian mixture density. Mixture models are a form of density model that consists of several components, usually Gaussian in nature. These functions of components are combined to give a multi-modal density. In this work, GMMs are developed to capture the information about each accident type, i.e. collision, rollover, and fall-off.

The number of Gausses in the mixture model is known as the number of components. They indicate the number of clusters in which data points are to be distributed. The number of components in each GMM is optimized based on the number of training data points. The components within each GMM capture finer level details among the feature vectors of each type of accident. Depending on the number of data points, the number of components may be varied in each GMM. If the GMM has a few components and is trained using a large number of data points, it may lead to more generalized clusters that fail to capture specific details related to each class. On the other hand, overfitting of the data points may happen, if too many components represent a few data points. Also, the complexity of the models increases if they contain a higher number of components.

The decision regarding the category of accident is taken based on its probability of coming from feature vectors of the specific model. Given a set of inputs, GMM refines the weights of each distribution through the expectation-maximization algorithm. Once a model is generated, conditional probabilities can be computed for test patterns (unknown data points). The ADC events are modeled using Gaussian PDFs, explained by the mean vector and the covariance matrix. For a feature vector x_t, the mixture density for an accident type is defined as the weighted sum of N component's Gaussian densities as

$$P(x_t \mid \Omega) = \sum_{i=1}^{N} W_i P_i(x_t) \tag{7}$$

where $P_i(x_t)$ is the component densities and w_i are the weights. For a D-dimensional feature vector, each component density is a D-variate Gaussian function, i.e.

$$P_i(x_t) = \frac{1}{(2\pi)^{\frac{D}{2}} \left|R_i\right|^{\frac{1}{2}}} e^{-\frac{1}{2}\left[(x_t - m_i)' \sum_i^{-1} (x_t - m_i)\right]} \tag{8}$$

where R_i is the covariance matrix for i^{th} component and m_i is a mean vector. The mixture weights satisfy the constraint

$$\sum_{i=1}^{N} W_i = 1 \tag{9}$$

Fig. 10 illustrates the overall methodology of a GMM-based accident classification system. The process is divided into two parts: (i) feature extraction phase and (ii) accident detection/classification phase. In the first phase, training models produce accident classification models by using a feature vector derived from the identified accident dataset by feeding the accident parameters to the GMM splitting and optimization block which uses the expectation-maximization algorithm. In the second phase, the testing (assessment) of the trained models is conducted using an unidentified accident dataset. Features of all unidentified accidents are provided to all trained models. The models then calculate the probability of unidentified vector features corresponding to a particular model. The most probable model is viewed as a potential accident case for that feature vector.

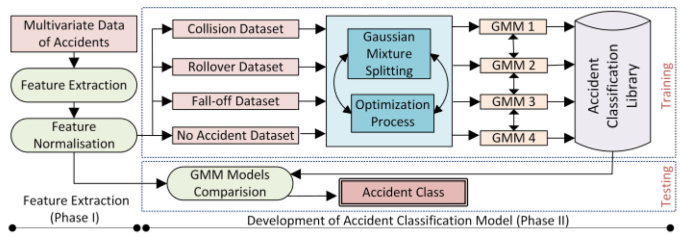

Fig. 10. Gaussian Mixture Model workflow for accident classification

B. Decision Tree (DT) based ADC Model

Decision Trees are prediction and classification tool with a tree-like structure, where a test is performed on an attribute at each node, each branch holds result of the test and each terminal node contains the class label. DTs are best suited for classification as they are strictly non-parametric, and does not require any information about the distribution of input parameters. DTs are capable of handling both numeric and categorical inputs, modeling nonlinear relationships between features and classes, allowing for missing values [40], [41]. DTs have an intuitive appeal because of the classification structure being explicit and easily interpretable.

The first in the development of a DT is tree growth. All input data is concentrated in the root node in the beginning. The dataset is then broken down into child nodes, with the application of a series of splitting variables (splitters). The DT algorithm measures the entropy, expected entropy and the gain in information to decide if an input variable must be selected as the splitter and if the node can be split further or not. Entropy is the measure of the amount of uncertainty of an event. Let M is an input variable for accident classification, with n distinct values. The entropy of the input variable is calculated by

$$E(M) = \sum_{i=1}^{n} -p_i \log_2 p_i \tag{10}$$

where p_i is the probability of taking a particular value and i is the number of options. The decision tree divides the input variable M into a number of subsets: $M_1, M_2, \ldots, M_n$, using a splitter. The expected unpredictability of the n outcomes of the input variable M is measured by using expected entropy (EE), defined by the equation

$$EE = \sum_{i=1}^{n} \frac{c_i}{c}(-p_i \log_2 p_i) \quad (11)$$

where EE is the expected entropy, c_i is the number of observations of the input variable in each subset $M_1, M_2..., M_n$, and c is total number of observations in parent node M.

The information gain *(I)* is the difference between expected entropy *(EE)* and actual entropy *E(M)*.

$$I = E(M) - EE \quad (12).$$

The node with information gain of 0 is treated as terminal node, which cannot be split further. A saturated tree is obtained with recursive application of previous steps. Five variables of our SNUSense database namely speed, ALA, roll, pitch, and change-in-altitude have been chosen as predictor variables, and collision, rollover, fall-off, and no-accident have been chosen as response classes (Fig. 11). Input dataset is divided into two datasets for training and testing. When the DT model fits the training data, the recursive algorithm of the DT model goes on to split the input data until it ends up with pure sets. Testing dataset is then used for testing the performance of trained model.

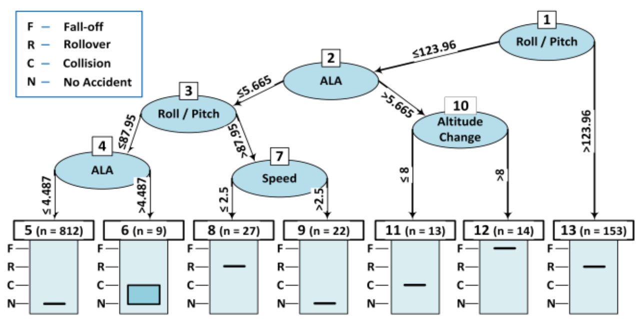

Fig. 11. Decision Tree for accident classification.

C. Naive Bayes (NB) based ADC Model

Another classification technique, NB, is used for ADC, which is based on Bayes' theorem. NB method is highly scalable and scales linearly with the number of predictors [42]. It is favored over other classification techniques because of its computational simplicity and its ability to be trained very fast [43]. It is robust to noisy data. The technique is based on the presumption that the predictor variables are independent, i.e. a particular feature present in a class is not related to the presence of other features.

The Naïve Bayes model contains the multiple input (or predictor) variables (change-in-altitude, pitch, roll, speed and ALA) and target variables (types of accident) as model outputs. Let T be the state or class of the target variable and vector $X = (x_1, x_2, ..., x_n)$ be the states of n input features. To estimate the value of T based on X, we need to calculate the conditional probability of T given X.

$$p(T \mid X) = \frac{p(X \mid T)p(T)}{p(X)} \quad (13)$$

where, *p(T)* and *p(X)* are the constants that are directly derived from the data. To obtain the value of *p(X|T)*, it is factorized as

$$p(X \mid T) = p(x_1, x_2, ..., x_n \mid T) = \prod_{i=1}^{n} p(x_i \mid T) \quad (14).$$

Combining Equations (13) and (14), we get

$$p(T \mid X) = \frac{p(T)}{p(X)} \prod_{i=1}^{n} p(x_i \mid T) \quad (15).$$

The conditional distribution of T given X is calculated from Equation (15). Value of target variable *p(T|X)* is classification outcome, which is the state of T with highest probability.

VIII. Results and Analysis

The performance of the ADC models based on GMM, NB and DT classifiers are discussed in this section. Outcomes have been obtained using the SNUSense database. All 1167 experimental observations of the SNUSense database is shuffled using random function, out of them 1050 (90%) observations are used for training the ADC models while 117 (10%) are used for testing. By an observation here we mean a data point that is the vector of values read from the sensors at a particular time during accident tests. The dataset is balanced and large enough to train and test the model. The classification performance of the proposed models is assessed using precision, recall, F1-score and receiver operating character (ROC) curve [44]. These quantities are described as following:

1. *Precision:* It is the ratio of the number of true positive (TP) predictions to all positive predictions. In this work, it represents the proportion of accident classifications that are actually right. i.e.

$$precision = \frac{true\_positive\_samples}{positively\_predicted\_samples} = \frac{TP}{(TP+FP)}$$

2. *Recall:* It is the ratio of number of true positives to all the actual positive records. It represents the proportion of accident classifications that can be predicted by model. i.e.

$$recall = \frac{true\_positive\_samples}{actual\_positive\_samples} = \frac{TP}{(TP+FN)}$$

3. *F1-score:* It takes both precision and recall into consideration. It is harmonic mean of precision and recall. i.e.

$$F1-score = 2*\left(\frac{precision*recall}{precision+recall}\right)$$

4. ROC Curve: The probability of detection i.e. true positive rate (TPR) is plotted as a function of the probability of false alarm i.e. false-positive rate (FPR) in the ROC curve for various threshold settings of the parameter. TPR (=TP/(TP+FN)) measures the proportion of actual positive incidents that are correctly identified. The FPR (=FP/(FP+TN)) is measured as the ratio between the number of falsely classified negative incidents as positive (false positives) and the overall number of TN incidents (irrespective of classification). TPR and FPR, both are different criteria which are compared in ROC curve. In our case, the area under the curve is an indicator of how well a function can distinguish a vehicle accident type with rest of the others.

The detection and classification performance of the NB technique is summarized in Table III. The mean of precision, recall, and F1-score for NB based classification model are 0.94, 0.95, and 0.95 respectively. The model performs best with collisions, where the F1-score is 0.97, which is followed by a rollover, no accident and fall-off events.

Accident classification performance of GMM models for the four accident classes using the SNUSense database has been computed. Using the 1050 observations and 16, 32, 64 and 128 components for building each GMM, performances are observed to be 89.75%, 87%, 95.5%, and 90.5%

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2020.3008896, IEEE Internet of Things Journal

respectively. During the training period, it is observed that the highest accident classification performance of 95.50% is achieved with 64 components and 100 iterations. If more than 64 components are used, the performance deteriorates.

TABLE III: DETECTION AND CLASSIFICATION PERFORMANCE USING NB

Accident Type	TP	FP	FN	TN	Precision	Recall	F1-score
Collision	32	1	1	83	0.96	0.98	0.97
Fall-off	27	2	2	86	0.92	0.94	0.93
Rollover	30	1	2	84	0.96	0.95	0.96
No-Acc.	22	1	1	93	0.94	0.95	0.94

Table IV is a summary of the accident detection and classification performance of the GMM technique on testing data. The average precision, recall and F1-score for the GMM based model are observed to be 0.91, 0.92 and 0.91, respectively. In the accident events, the model performs best with collisions with an F1-score of 0.92, followed by fall-off and rollover. The non-occurrence of a road accident is reported with an F1 value of 0.94.

TABLE IV: DETECTION AND CLASSIFICATION PERFORMANCE USING GMM

Accident Type	TP	FP	FN	TN	Precision	Recall	F1-score
Collision	31	3	2	81	0.91	0.94	0.92
Fall-off	26	3	3	85	0.90	0.91	0.90
Rollover	28	4	4	81	0.88	0.87	0.87
No-Acc.	22	1	1	93	0.94	0.94	0.94

Table V describes the detection and classification performance of the DT model. The model performs best with rollover events with an F1-score of 0.92, followed by fall-off and collision. The mean precision, recall and F1-score of the DT model are observed to be 0.88, 0.89 and 0.88 respectively.

TABLE V: DETECTION AND CLASSIFICATION PERFORMANCE USING DT

Accident Type	TP	FP	FN	TN	Precision	Recall	F1-score
Collision	29	4	4	80	0.86	0.87	0.86
Fall-off	26	3	3	85	0.90	0.89	0.89
Rollover	30	3	2	82	0.92	0.93	0.92
No-Acc.	20	3	3	91	0.86	0.88	0.87

It is evident that the Naïve Bayes model is the best performing model with an average F1-score of 0.95, which is followed by the GMM and DT models with an F1-score of 0.91 and 0.88 respectively.

ROC curves of NB-, GMM-, and DT-based models are plotted in Fig. 12, 13, and 14, respectively. Actually, ROC curve is a graphical plot which demonstrates a binary classifier system's diagnostic capability. Because our problem is a multi-classification problem, each ROC curve of a particular model is drawn by using One-versus-Rest scheme where each accident scenario is plotted against all other accident types, which makes it binary classification problem for ROC curves of every accident types. With the help of positive and negative outcomes of ADC models, we have used binary logistic regression and saved the prediction probabilities. Then for generating the multivariable ROC curve for a particular accident class, saved prediction probabilities as the test variables are utilized.

To train, test, and identify the best ADC model among NB-, GMM- and DT-based models, first the "*scikit-learn*" Python library and SNUSense dataset are used on a separate computer. After identifying that the NB-based model is the best ML model over GMM and DT for our ADC problem, the NB-based model is implemented in the SNUSense application. We have used the "JNBC" Java library [45] for Naïve Bayes classification, which is provided by NamSor.com. Although, every ML algorithm is hard to implement in smartphones because of their processing toll and availability of a suitable library. For this purpose, the concept of IoT server is used in proposed architecture to implement the ML algorithm on the server-side and to establish publish/subscribe communication.

In case of rollover and fall-off accidents, RC cars can mimic the real car accidents completely. As far as the intensity of deceleration generated due to the impact is concerned, it may differ due to the build, size, and weight of a real car, thus deceleration threshold could be set accordingly while training the system for real cars.

IX. CONCLUSION AND FUTURE WORK

The proposed detection system works frequently whether there is an accident or not and reports the incident to predefined emergency services and family in case of an accident. The system classifies accidents into four classes i.e. collisions, rollovers, fall-offs, and no accidents, so that the best possible rescue operations can be undertaken. Five training variables namely change-in-altitude, pitch, roll, speed and ALA are used as input variables to train and test the system. The proposed ADC system uses smartphone sensors, and Sensordrone sensors to measure the values of model variables. The system can be retrofitted in any type of vehicle.

As far as accuracy of proposed ADC system is concerned, three different GMM-, NB-, and DT-based classification models are evaluated and compared to determine the most accurate ADC model. The NB model is found to outperform others with an average F1-score of 0.95. It is the most accurate for collision with an F1-score of 0.97, followed by rollover, no-accident and fall-off events respectively.

If someone wants to reduce the time of automatic notification after the incident or compare the system with other automatic notification systems, every delay should be as short as possible whether it is algorithm's execution time or time taken in notification. But this work focuses on reducing overall reporting time using technology, compared to just manual reporting or non-reporting of an incident. To achieve

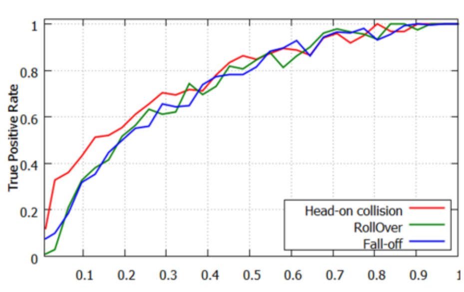

Fig. 12. ROC curve for NB.

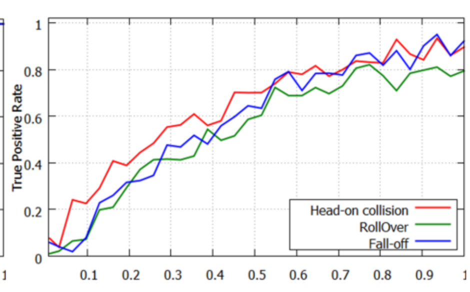

Fig. 13. ROC curve for GMM.

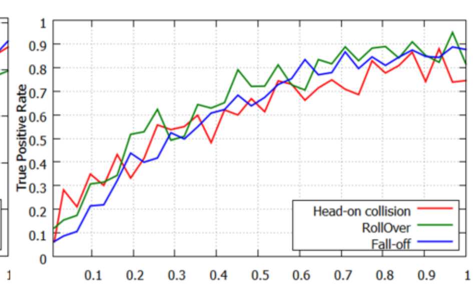

Fig. 14. ROC curve for DT.

this, we have tried to reduce the time taken for notification after accident detection by implementing machine learning algorithms on the smartphone itself rather than transmitting data of all sensors after every 10ms.

The system is low-cost in comparison to factory fitted systems because they substantially increase the costs of the cars. According to [46], there were more than 3.2 billion smartphone users in the world in 2019. Hence, we have used a smartphone and its built-in sensors to develop our system. Apart from the smartphone, only a barometric altimeter and a 4G internet connection is needed to implement the system.

To the best of our knowledge, our system is the only one-of-a-kind system that classifies accidents as collision, rollover, fall-off, and no-accident. Although system is highly accurate and have several advantages over other systems, it also has some limitations, i.e. (a) System needs continuous internet connection to send the emergency alerts. (b) Placement of the smartphone would be predefined and user cannot put his smartphone at any other place such as pocket, bag, etc., as otherwise chances of FP (in case of no accident) and FN (in case of an accident) would be very high. (c) If smartphone gets ejected outside the vehicle, or hardware setup breakdowns, results may be impacted, and system may fail.

As a future work, we intend to add another accident class i.e. fire/explosion, to our ADC model. To identify the most appropriate classification model for our ADC system, we are going to implement some validation schemes such as K-fold cross validation, and we will also consider the time taken by the classification algorithm. In this work, both Android apps are function oriented, but in future we are also going to merge them in a single app having a user-friendly and interactive interface to make a unified solution for our ADC system.

REFERENCES

[1] WHO, “World Health Statistics 2014: A Wealth of Information of Global Public Health,” Geneva, 2014. [Online]. Available: https://apps.who.int/iris/handle/10665/112739.

[2] WHO, “Global Status Report on Road Safety 2018: Summary,” Geneva, 2018. [Online]. Available: https://www.who.int/violence_injury_prevention/road_safety_status/2018/English-Summary-GSRRS2018.pdf.

[3] E. B. Lerner and R. M. Moscati, “The Golden Hour: Scientific Fact or Medical ‘Urban Legend’?,” *Acad. Emerg. Med.*, 2001, doi: 10.1111/j.1553-2712.2001.tb00201.x.

[4] R. Sánchez-Mangas, A. García-Ferrrer, A. De Juan, and A. M. Arroyo, “The probability of death in road traffic accidents. How important is a quick medical response?,” *Accid. Anal. Prev.*, 2010, doi: 10.1016/j.aap.2009.12.012.

[5] M. A. Razzaque, M. Milojevic-Jevric, A. Palade, and S. Cla, “Middleware for internet of things: A survey,” *IEEE Internet Things J.*, 2016, doi: 10.1109/JIOT.2015.2498900.

[6] G. Ponte, G. A. Ryan, and R. W. G. Anderson, “An estimate of the effectiveness of an in-vehicle automatic collision notification system in reducing road crash fatalities in South Australia,” *Traffic Inj. Prev.*, 2016, doi: 10.1080/15389588.2015.1060556.

[7] Y. Chung and W. W. Recker, “A Methodological Approach for Estimating Temporal and Spatial Extent of Delays Caused by Freeway Accidents,” *IEEE Trans. Intell. Transp. Syst.*, 2012, doi: 10.1109/tits.2012.2190282.

[8] F. Aloul, I. Zualkernan, R. Abu-Salma, H. Al-Ali, and M. Al-Merri, “IBump: Smartphone application to detect car accidents,” *Comput. Electr. Eng.*, 2015, doi: 10.1016/j.compeleceng.2015.03.003.

[9] F. Bhatti, M. A. Shah, C. Maple, and S. Ul Islam, “A novel internet of things-enabled accident detection and reporting system for smart city environments,” *Sensors (Switzerland)*, 2019, doi: 10.3390/s19092071.

[10] A. Shaik *et al.*, “Smart Car: An IoT Based Accident Detection System,” in *2018 IEEE Global Conference on Internet of Things, GCIoT 2018*, 2019, doi: 10.1109/GCIoT.2018.8620131.

[11] S. Sharma and S. Sebastian, “IoT based car accident detection and notification algorithm for general road accidents,” *Int. J. Electr. Comput. Eng.*, 2019, doi: 10.11591/ijece.v9i5.pp4020-4026.

[12] E. K. Priya et al., “IoT Based Vehicle Tracking and Accident Detection System,” *Int. J. Innov. Res. Comput. Commun. Eng.*, vol. 5, no. 3, 2017.

[13] B. K. Dar, M. A. Shah, S. U. Islam, C. Maple, S. Mussadiq, and S. Khan, “Delay-Aware Accident Detection and Response System Using Fog Computing,” *IEEE Access*, 2019, doi: 10.1109/ACCESS.2019.2910862.

[14] B. Fernandes, M. Alam, V. Gomes, J. Ferreira, and A. Oliveira, “Automatic accident detection with multi-modal alert system implementation for ITS,” *Veh. Commun.*, 2016, doi: 10.1016/j.vehcom.2015.11.001.

[15] P. Fabian, A. Rachedi, and C. Guéguen, “Programmable objective function for data transportation in the Internet of Vehicles,” *Trans. Emerg. Telecommun. Technol.*, 2020, doi: 10.1002/ett.3882.

[16] A. Rachedi and H. Badis, “BadZak: An Hybrid Architecture Based on Virtual Backbone and Software Defined Network for Internet of Vehicles,” in *IEEE International Conference on Communications*, 2018, doi: 10.1109/ICC.2018.8422759.

[17] T. Mekki, I. Jabri, A. Rachedi, and M. Ben Jemaa, “Vehicular cloud networking: Evolutionary game with reinforcement learning-based access approach,” in *International Journal of Bio-Inspired Computation*, 2019, doi: 10.1504/IJBIC.2019.097730.

[18] D. Acharya, V. Kumar, and G. M. Gaddis, “A Mobile System for Detecting and Notifying Vehicle Rollover Events,” in *15th International Conference on Advanced Computing and Communications (ADCOM 2007)*, Dec. 2007, pp. 268–275, doi: 10.1109/ADCOM.2007.102.

[19] H. A. Ibrahim, A. K. Aly, and B. H. Far, “A system for vehicle collision and rollover detection,” in *Canadian Conference on Electrical and Computer Engineering*, 2016, doi: 10.1109/CCECE.2016.7726720.

[20] J. Smolka and M. Skublewska-Paszkowska, “A method for collision detection using mobile devices,” in *Proceedings - 2016 9th International Conference on Human System Interactions, HSI 2016*, 2016, doi: 10.1109/HSI.2016.7529620.

[21] S. Sadek, A. Al-Hamadi, B. Michaelis, and U. Sayed, “Real-time automatic traffic accident recognition using HFG,” in *Proceedings - International Conference on Pattern Recognition*, 2010, doi: 10.1109/ICPR.2010.817.

[22] F. Felisberto, F. Fdez-Riverola, and A. Pereira, “A ubiquitous and low-cost solution for movement monitoring and accident detection based on sensor fusion,” *Sensors (Switzerland)*, 2014, doi: 10.3390/s140508961.

[23] I. G. Damousis and D. Tzovaras, “Fuzzy fusion of eyelid activity indicators for hypovigilance-related accident prediction,” *IEEE Trans. Intell. Transp. Syst.*, 2008, doi: 10.1109/TITS.2008.928241.

[24] Z. Zhang, Q. He, J. Gao, and M. Ni, “A deep learning approach for detecting traffic accidents from social media data,” *ArXiv*, vol. abs/1801.0, 2018.

[25] S. Moulik and S. Majumdar, “FallSense: An Automatic Fall Detection and Alarm Generation System in IoT-Enabled Environment,” *IEEE Sens. J.*, 2019, doi: 10.1109/JSEN.2018.2880739.

[26] D. Lohani and D. Acharya, “Real time in-vehicle air quality monitoring using mobile sensing,” in *2016 IEEE Annual India Conference, INDICON 2016*, 2017, doi: 10.1109/INDICON.2016.7839099.

[27] National Marine Electronics Association et al, *NMEA 0183--Standard for interfacing marine electronic devices*. NMEA, 2002.

[28] Y. Lee, H. Yeh, K. H. Kim, and O. Choi, “A real-time fall detection system based on the acceleration sensor of smartphone,” *Int. J. Eng. Bus. Manag.*, 2018, doi: 10.1177/1847979017750669.

[29] J. Kendall and K. A. Solomon, “Air Bag Deployment Criteria,” *Forensic Exam.*, 2014.

[30] H. Ye, K. Dong, and T. Gu, “HiMeter: Telling you the height rather than the altitude,” *Sensors (Switzerland)*, 2018, doi: 10.3390/s18061712.

[31] N. Kumar, A. Barthwal, D. Acharya, and D. Lohani, “Modeling IoT Enabled Automotive System for Accident Detection and Classification,” in *IEEE Sensors Applications Symposium (SAS)*, 2020.

[32] V. Kubelka and M. Reinstein, “Complementary filtering approach to orientation estimation using inertial sensors only,” in *Proceedings - IEEE International Conference on Robotics and Automation*, 2012, doi: 10.1109/ICRA.2012.6224564.

[33] S. Hajdu, S. T. Brassai, and I. Szekely, “Complementary filter based

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2020.3008896, IEEE Internet of Things Journal

sensor fusion on FPGA platforms," in *Proceedings - 2017 International Conference on Optimization of Electrical and Electronic Equipment, OPTIM 2017 and 2017 Intl Aegean Conference on Electrical Machines and Power Electronics, ACEMP 2017*, 2017, doi: 10.1109/OPTIM.2017.7975076.

[34] M. Iyoda, T. Trisdale, R. Sherony, D. Mikat, and W. Rose, "Event Data Recorder (EDR) developed by Toyota Motor Corporation," *SAE Int. J. Transp. Saf.*, 2016, doi: 10.4271/2016-01-1495.

[35] GP TOYS, "Foxx S911 RC Truck," 2020. https://g-p.hk/gptoys-foxx-s911.html (accessed Apr. 05, 2020).

[36] Google Developers, "Google Firebase," 2020. https://firebase.google.com/ (accessed Apr. 05, 2020).

[37] Batelle Memorial Institute, "Crashworthiness research of prototype hydrogen fuel cell vehicle: Task order 7 report (Report No. DOT HS 812 112)," Washington, DC, 2015.

[38] K. S. Rao and S. G. Koolagudi, "Robust Emotion Recognition using Combination of Excitation Source, Spectral and Prosodic Features," in *Springer Briefs in Electrical and Computer Engineering*, New York: Springer, 2013, pp. 85–94.

[39] Y. Sun and B. Yuan, "Hierarchical GMM to handle sharp changes in moving object detection," *Electron. Lett.*, 2004, doi: 10.1049/el:20040552.

[40] Z. Zheng, P. Lu, and D. Tolliver, "Decision tree approach to accident prediction for highway-rail grade crossings: Empirical analysis," *Transp. Res. Rec.*, 2016, doi: 10.3141/2545-12.

[41] J. Zheng, S. Yang, X. Wang, X. Xia, Y. Xiao, and T. Li, "A Decision Tree Based Road Recognition Approach Using Roadside Fixed 3D LiDAR Sensors," *IEEE Access*, 2019, doi: 10.1109/ACCESS.2019.2912581.

[42] T. Šingliar and M. Hauskrecht, "Learning to detect incidents from noisily labeled data," *Mach. Learn.*, 2010, doi: 10.1007/s10994-009-5141-7.

[43] B. Yang, Y. Lei, and B. Yan, "Distributed Multi-Human Location Algorithm Using Naive Bayes Classifier for a Binary Pyroelectric Infrared Sensor Tracking System," *IEEE Sens. J.*, 2016, doi: 10.1109/JSEN.2015.2477540.

[44] Z. Omary and F. Mtenzi, "Machine Learning Approach to Identifying the Dataset Threshold for the Performance Estimators in," *Int. J. Infonomics*, 2010, doi: 10.20533/iji.1742.4712.2010.0034.

[45] E. Carsenat, "Java-Naïve-Bayes-Classifier-JNBC," *NamSor SAS*, 2020. https://github.com/namsor/Java-Naive-Bayes-Classifier-JNBC.

[46] S. O'Dea, "Smartphone users worldwide 2016-2021," 2020. https://www.statista.com/statistics/330695/number-of-smartphone-users-worldwide/ (accessed Apr. 05, 2020).

Nikhil Kumar is pursuing PhD in Computer Science and Engineering from Shiv Nadar University, India.

His research interests are context aware computing, wireless sensor networks, vehicular accident detection and classification modeling, and Internet of Things.

Debopam Acharya earned his PhD from the University of Missouri-Kansas City, Missouri, USA.

He is currently Professor and Dean of School of Computing at DIT University, Dehradun, India. His research interests are in the areas of Internet of Things, Mobile Sensing and Smart Environment. He has worked in the Location Based Services group at Garmin Inc., Olathe, Kansas, USA and tenure track faculty at Georgia Southern University, Statesboro, GA, USA.

Dr. Acharya's research has been funded by several public and private sector organizations like BlackBerry, Dell, and the Ministry of Electronics and IT, Govt. of India.

Divya Lohani has received her PhD from Indian Institute of Information Technology, Allahabad, India.

She is serving as an Assistant Professor in the Department of Computer Science & Engineering at Shiv Nadar University, India. Her research interests include Internet of Things, sensor networks, sensor data analytics, context aware computing, mobile computing and cybersecurity.

Dr. Lohani is a member of the IEEE.