Intelligent Accident Detection Classification using Mobile Phones

Conference Paper · January 2018

DOI: 10.1109/ICOIN.2018.8343170

Imran A. Zualkernan
American University of Sharjah
193 PUBLICATIONS **3,612** CITATIONS
SEE PROFILE

Fadi A. Aloul
American University of Sharjah
166 PUBLICATIONS **5,172** CITATIONS
SEE PROFILE

IEEE International Conference on Information Networking (ICOIN), Chiang Mai, Thailand, January 2018.

Intelligent Accident Detection Classification using Mobile Phones

Imran A. Zualkernan, Fadi Aloul, Fayiz Basheer, Gurdit Khera, and Shruthi Srinivasan

Department of Computer Science & Engineering
American University of Sharjah, UAE

***Abstract*—Road accidents are one of the leading causes of mortality. While most accidents merely affect the exterior of the cars of the drivers involved, some of them have led to serious and fatal injuries. It is imperative that the Emergency Medical Services (EMS) are given as much information about the crash site as possible before their arrival at the scene. In this paper, a mobile phone application is developed that, when placed inside a car, intelligently classifies the type of accident it is involved in and notifies the EMS team of this classification along with the car's GPS location. The classification mechanism is built through a collection of data sets from a simulation of three types of collisions, which creates a knowledge base for an artificial intelligence-based classifier software. The experimental setup for data collection and the functionality of the mobile phone application called 'Crash Detect' are explored.**

Keywords*—Accident Detection, Machine intelligence, Mobile phones, Sensors

I. Introduction

Road accidents have been one of the top contributors towards human fatality for decades. Traffic accidents are the leading cause of mortality for individuals aged 19 to 25 around the world [1], and the second major cause of deaths in the UAE [2]. A series of events takes place from the time an accident has taken place until the patient has fully recovered. This paper proposes a method to increase the efficiency of one of the most vital parts of this sequence called the 'emergency response.' Emergency response is the process of medical personnel's first response to the accident and consists of medical personnel arriving at the scene of the accident. In [3], the authors describe the importance of knowing the mechanics of a crash while paramedics attend to a crash site. Similarly, in [4], the authors also stress the importance of understanding the kinematics of an accident. Kinematics is an analytical processing of the scene of a crash to determine the injuries most likely to have been caused by the forces and movements involved. Such understanding may be employed to identify stable patients with seemingly no injury who might still carry the risk of later suffering severe conditions due to the high intensity of the accident they were involved in. This method also helps pre-hospital healthcare professionals, or paramedics, carry out evaluations and identify injuries in cases where anatomical and physiological criteria for severity are inconclusive. This process also assists the paramedics in deciding which hospital is best equipped to handle the severity of the wounds incurred, prepare for extrication if necessary and gather other resources to conduct a complete and thorough evaluation of the victim.

The research presented in this paper is motivated by the fact that emergency medical service personnel would perform better if equipped with the knowledge of the kind of accident they are dealing with before arriving at the accident scene.

Several technology-based solutions have been proposed to reduce fatalities in road accidents during emergency response. For example, several vehicles now come with built in systems that detect crashes and notify Emergency Management Systems (EMS) within seconds of the incident. The Automatic Crash Notification (ACN) systems built into some vehicles provide paramedics with information regarding a crash before their arrival at the scene. Such systems are typically available in modern cars and are activated when the airbag is deployed (e.g., Ford SYNC 911 Assist system) or the emergency fuel pump shut-off is activated. In some instances, the system is activated even without the deployment of an airbag, using sensors placed around the vehicle (e.g., the GM OnStar system). Some of these systems send a distress message to the local police with the location of the victim when an accident is suspected [5]. In some cases, the car's occupants are also connected via phone to local EMS hotlines until help arrives.

Although built-in ACN systems offer a viable solution in notifying EMS personnel, there are a few drawbacks associated with such systems. Firstly, these systems are only available in newer models of cars only. For example, the Toyota company offers ACN in models from the year 2010 onwards. Secondly, most of these systems are on a subscription basis; monthly payments are required to order to keep the service running. Therefore, cost is a major concern when it comes to using ACN systems. In some cases, the occupants of the vehicle are not conscious enough after the accident to communicate their status to the EMS personnel via the phone connection established by the system. Furthermore, these systems could be rendered obsolete should the manufacturing company introduce a major change in their communication strategy.

These potential drawbacks for current ACN systems call for a notification system that is cost-effective, portable, and can be extended to situations that do not involve vehicular collisions, places the burden of communication away from the injured party, and detects the kind of collision the victim is involved in.

This paper proposes such a notification system implemented as an application on the driver's mobile phone. A mobile phone is the good platform for such a system because most drivers today carry mobile phones. In addition, most mobile phones also come with built-in accelerometers and gyroscope sensors that can be used to help measure crash kinematics in case of an accident.

The system presented in this paper is a mobile phone application called 'Crash Detect' that detects collisions and classifies the collision into one of two types: (1) car to non-deformable or rigid object, and (2) car to human collision, and notifies local police, EMS and victim's emergency contacts of the location and type of crash via SMS. The system also incorporates handling false positives like the phone dropping, sharp turns or uneven road conditions.

The remainder of the paper is organized as follows. The next section describes previous work in detecting accidents using mobile phones. Section III describes the design of the system including an evaluation of the machine learning algorithms used. Section IV shows a comparison between algorithms and experimental results. Section V presents a description of the developed mobile application. Finally, Section VI concludes the paper.

II. LITERATURE REVIEW

This section provides a brief summary of works related to using mobile phones for detecting accidents. In 2010, a group of researchers from Vanderbilt University proposed the mobile phone application *WreckWatch* [6]. This application keeps track of accident/event data by recording the path, speed, forces of acceleration on a vehicle leading up to the accident. The research was conducted with the assumption that the phone is inside the user's pocket at all times, and therefore any forces applied to the phone would have been applied to the user. This system is an Android application on the client and Java/MySQL and uses the Spring Framework on the server. This paper provides an in-depth analysis regarding their client server architecture and the use of what they refer to as 'on board' sensors; accelerometer and GPS sensors were used to detect collisions. False positives were prevented by using contextual information, speed and acceleration filters. Other challenges addressed in this research include excessive power consumption, possible destruction of the phone, and determining whether or not the user is inside the vehicle.

Another implementation of a similar system was reported in [7]. The proposed system uses onboard sensors to collect dynamic data witnessed in vehicular motion and uses this raw data to categorize different collision events. The categorization is done through using different machine learning algorithms. The system is designed as a life-long learning system that records the driver's driving style and may be used to detect the driver's state (sleepy, intoxicated, etc).

In a similar system [8], the authors devised a system that was 84% accurate in classifying accidents into different categories. Their work elaborates on the methods used in classifying car accidents into three classes; head-on collision, collision to a barrier, and collision to wall.

Another system called *iBump* [9] continuously detects if an accident has taken place using the built-in accelerometer of the mobile phone. In case of an accident, the application sends an SMS with the victim's GPS coordinates to the listed emergency contacts and the GSM server. The research was conducted based on data taken from a simulation of accidents using a mechanical model. The simulation was done only for the case where a car collides with a rigid stationary object. The accident was classified based on severity.

Finally, as [10] points out that implementation of the crash detection problem using a mobile device should not only distinguish between various types of accidents but should try to account for obvious false positives like dropping the phone, shaking, sudden breaks on a physical apparatus and data collected in actual vehicles engaged in behavior like sharp turns etc.

III. TECHNICAL APPROACH

This section discusses the process involved in implementing the proposed solution. The proposed system is an Android application that detects a road accident using the accelerometer and gyroscope in mobile phones. Based on sensor data, the system classifies the accident into one of two classes and notifies local police, EMS personnel and emergency contacts. The system was developed by simulating real life accidents in a controlled environment and recording data with sensors. Features were then extracted from this data and then used to train and test a number of algorithms. The performance of these algorithms was evaluated based on their ROC curves, precision, recall and F-Measures. The best performing algorithms were then integrated into the Android application for further testing. Each of the steps are described next.

A. Experimental Setup and Accident Simulation

The objective of the experimental setup was to generate forces involved in real life car accidents and collect the accelerometer and gyroscope signatures for these situations. The force involved in a high intensity accident is approximately $40g$ where g is the acceleration due to gravity. In order to generate this force, the apparatus shown in Figure 1 was used where springs were attached to a vehicle of 2 kg payload comprising of sensors and pulled to a distance of 1.5 meters and then released to simulate a collision. The spring constant k of these springs was calculated to ensure that a force of approximately $40g$ was generated upon impact.

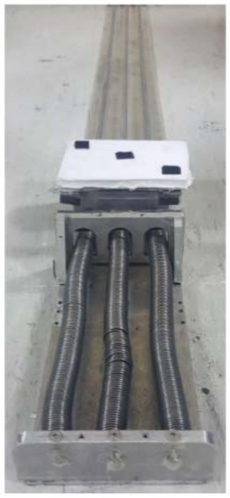


Fig 1. Apparatus to simulate the $40g$ crash

In order to collect accelerometer and gyroscopic data, a sensor box mimicking a mobile phone placed was placed on the apparatus as the payload. As show in Figure 2, this data collection sensor box consisted of the following components:

- LilyPad Arduino - microcontroller used to store data collected by sensors
- 9 Degrees of Freedom Sensor Stick - contains an accelerometer (± 16g), gyroscope (± 2000°/s)
- Accelerometer ADXL377 (± 200g) - to measure accelerations that exceed 16g
- SD card breakout to store readings to an SD card
- Power Supply – Polymer Lithium Ion battery with 7V nominal output voltage
- Power Supply Stick – to bring down the voltage from 7V to 3.3 volts.

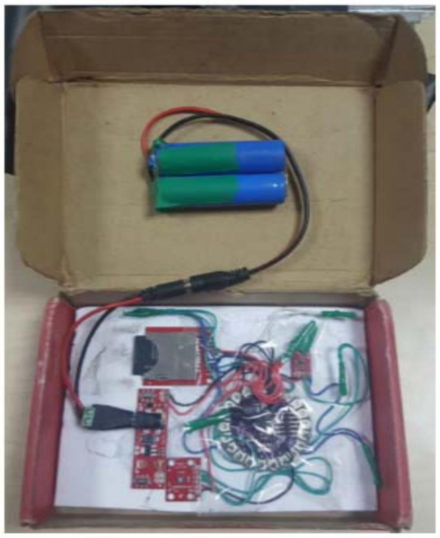


Fig 2. Sensor box to collect accelerometer and gyroscopic sensor data

B. Data Collection and Cleaning

Simulations were performed to collect sensor readings for data that belonged to each of the classes shown in Table 1.

TABLE 1. CLASSES OF DATA

Scenario Type	Collision Scenario	No. Samples
Accident	Car to rigid wall	80
	Car to human being	80
False Positive	Phone being dropped	50
	Sharp Turns	28
	Driving on unpaved road	30

As Table 1 shows, two types of conditions were simulated; actual accident and false positives. Two types of accidents were simulated; vehicle hitting a rigid object and vehicle hitting a person. The vehicle hitting a person was simulated by using a hot water bag filled with water. Similarly, three types of false positives were considered; phone being dropped, driving on unpaved road or taking sharp turns. Data for accidents were collected using the apparatus show in Figure 1 while the data for false positives was collected in a real car. The sensor box was used in each case to collect the data. Sensor readings were taken every five milliseconds in each instance. The readings were stored and then retrieved from the SD card installed in the sensor box. Each text file was then individually cleaned; approximately 700 readings were recorded for each sample from which the point of impact was manually found and extracted. A window of 32 readings around the crash was then used as the actual data showing a particular condition (e.g., rigid crash). This served as the raw data for training the system.

C. Feature Extraction

The next step in the process is the extraction of features from the raw data. For example, in [7], mean, standard deviation, entropy, energy and maximum peaks from accelerometer sensor readings were used as features and then directly used for attribute selection. Similarly, in [11] feature extraction was carried out incorporating the discriminating ability of features. The measurements used to evaluate the performance of the features were the Bhattacharyya distance, Fisher's discriminant ratio and divergence. The conclusion was that classification with extracted features yielded a higher performance accuracy when compared to classification with raw data alone.

In [12], Discrete Wavelet Transforms (DWT) based lossy compression was used to overcome challenges of storage and computation time. Another study [13] also illustrated how DWTs were useful in defining new sets of features used in classification and similarity search applications from wavelet coefficients. These usually result in better defined features due to the reduction in noise or irrelevant data, increasing the accuracy of classification and similarity search. The authors in [11] also detail the need to identify appropriate features that differentiate textures (or in our case, signals) for classification. The authors found the application of DWT to a set of statistical features extracted from raw data provided a precise and unifying framework for the analysis and characterization of a signal at different scales.

Based on prior research, the research reported here applied Discrete Wavelet Transforms (DWT) to the selected features to extract time-frequency information and to achieve noise suppression. A wavelet is an orthogonal function that is applied to an infinite group of data. The following equation describes a typical DWT decomposition:

$$DWT(m,k) = \frac{1}{a} \sum_{n=0}^{N-1} s(n) g\left(\frac{k-b}{a}\right) \quad (1)$$

Where *s(n)g* is the original signal, *N* is the number of samples in the windowed signal (in our case N= 32), *m* is the decomposition level index while *a* and *b* are scaling and translation parameters respectively. The DWT can be interpreted as a multi-stage filter banks with HP (high pass resulting in approximate coefficients) and LP (low-pass resulting in detail coefficients) filters performing series dilations [14].

At each level the approximate/detail coefficients represent a filtered signal that spans only half the frequency of the band. This improves the frequency resolution as the frequency uncertainty is reduced by half. A schematic representation of the working of DWT is shown in Figure 3 where A1, A2 and A3 represent sequences of approximate coefficients and D1, D2 and D3 represent sequences of detailed coefficients.

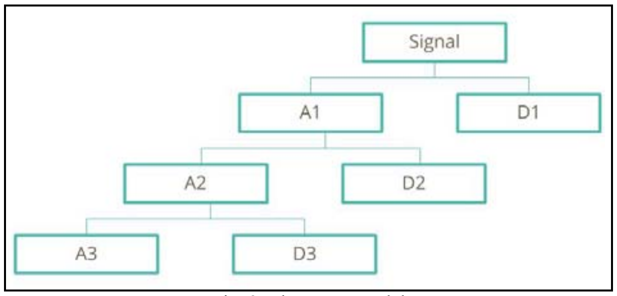


Fig. 3. The DWT model

The coefficients produced by the application of DWT however are useable only up until a certain level of decomposition. Lei et. Al. [14] demonstrated the impact of choosing the appropriate level of decomposition (DL) in DWT on its performance. They quantified the degree of sparseness in wavelet coefficients by evaluating the percentage of zero/near-zero coefficients among the entire transformed coefficients. To determine the number of levels of DWT to be performed, the sparseness of the coefficients in each level was calculated. The percentage of zero/near-zero coefficients among the transformed data was evaluated. The following equation was used in this calculation:

$$sp' = \frac{N_O}{N-1} \qquad (2)$$

where *sp'* represents the sparseness, *N* is the window size (32 in this case) and *No* is the number of zero/near zero coefficients at that particular level. This was done to constrain the value of sparseness between [0, 1]. The average of the sparseness of wavelet coefficients in different classes of data collected was plotted against their respective level. The graph so obtained in Figure 4 shows that sparseness of the transformed data diminishes and becomes almost saturated as the decomposition level reaches 4. Therefore, decomposition was stopped at level 3.

After the DWT encoding, following prior research, Mean, Standard Deviation, Kurtosis, Linear Weighted Average, Quadratic Weighted Average, and Skewness was calculated for each approximate (i.e., A1, A2, and A3) and detailed coefficient (i.e., D1, D2, and D3) sequence as the primary features for each 32-point raw data sequence collected earlier for each scenario (i.e., Rigid Crash). This resulted in one feature vector per instance which was used in the next stage of training classifiers.

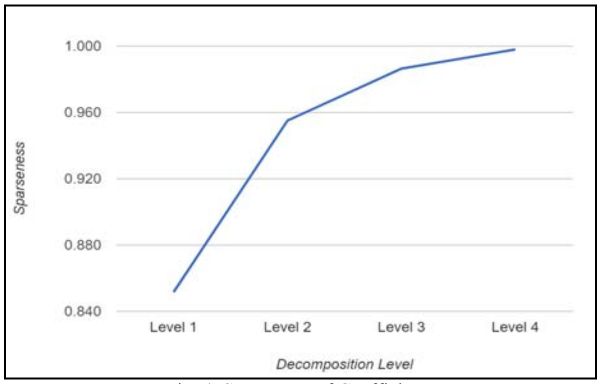


Fig. 4. Sparseness of Coefficients

D. Classification Algorithms

Various modelling techniques have been explored in classifying accident data collected from mobile devices. For example, the *iBump* application used a hybrid of Hidden Markov Models (HMM) and Dynamic Time Warping (DTW) for classifying accidents. The Hidden Markov Model is a modeling tool that can be employed for modeling and analyzing time series with spatial or temporal variability [9]. This technique has been employed in various speech and gesture recognition systems. The authors in [15], described in detail the application of HMM for very large vocabulary speech recognition. The sensing process is basically collection of data from the mobile phone's sensors. As today's mobile phones are only sensitive to forces between -4*g* and + 4*g*, the sensor data is clipped from its original value into this range. *iBump* used Weighted Moving Average (WMA) to clean the data. The data was then passed into a Vector Quantization Unit in order to translate the three dimensional data into a single dimensional sequential vector. These vectors were then used to train the HMM model and as data for accident classification.

Dynamic Time Warping (DTW) is another algorithm used in machine learning. This algorithm measures similarity between two temporal sequences in a time series that may have different speeds. For example, this algorithm used gesture recognition based on accelerometer data to create a wireless system for cricket training [16]. In the *iBump* system, this algorithm was used to detect accidents alongside HMM due to low false positive rates of HMM and an inability to tell different levels of severity apart. Baum-Welch algorithm was used for HMM training. This algorithm automatically estimates the parameters of a HMM and determines the probability of occurrence of a test set. It then uses this test set as a learning mechanism for the model.

$$\alpha + \beta = \chi . \quad (1) \quad (1)$$

A number of traditional classification algorithms have also been used to solve similar problem [7]. These methods included Bayes Network, Random Forests, Logistic Regression and Radial Basis Function. The algorithms were evaluated based on their accuracy, F-Measure and low error on misclassification. The evaluation revealed the random forest classifier to be the best one with an accuracy of 78.44% and an F-Measure of approximately 0.8. The same platform was used by the researchers in [7] whilst testing the following algorithms: REPTree, Random Forest, JRIP and RBF Classifier of which REPTree outperformed the rest with an evaluation performance accuracy of 84.3%.

Based on prior research, this paper applied the following classification algorithms for classifying the WDM-coded sensor data:

- Naïve Bayes
- Bayesian Networks
- Logistic Regression
- Random Forest
- REPTree
- J48

The Wrapper Sub Set Selection feature selection method was used for each classifier which resulted in different sub-set of attributes for training each of the classifiers tested. The attributes selected in both the aforementioned methods were trained and tested after applying the SMOTE filter for underrepresented classes. A 10-fold cross-validation was used to train and test each algorithm.

IV. RESULTS AND EVALUATION

Results for the top three best performing classifiers are presented in this section.

A. *Naïve Bayes*

The Naïve Bayes algorithm performed best in terms of the 'Sharp Turn' and 'Unpaved Road' classes with a True Positive rate of over 97%. However, this algorithm significantly misclassified the Phone drop class with only 60% accuracy. There were also a number of instances where an accident case was classified as a non-accident case. The complete performance of the Naïve Bayes classifier is displayed in Table 2 below.

TABLE 2. PERFORMANCE OF THE NAÏVE BAYES CLASSIFIER

TP Rate	FP Rate	Precision	Recall	F-Measure	ROC Area	Class
0.7	0.083	0.667	0.7	0.683	0.919	Wall Crash
0.888	0.068	0.755	0.888	0.816	0.96	Human Crash
0.6	0.041	0.667	0.6	0.632	0.886	Phone Drop
0.972	0	1	0.972	0.986	1	Sharp Turn
0.981	0.003	0.981	0.981	0.981	0.996	Unpaved Road

B. *Logistic Regression*

Logistic Regression performed slightly better than the Naïve Bayes. This algorithm did a relatively better job of differentiating between accident and non-accident cases. For example, the Phone Drop was classified correctly 74% of the time. The overall performance of the Logistic Regression classifier is shown in Table 3 below.

TABLE 3. PERFORMANCE OF THE LOGISTIC REGRESSION CLASSIFIER

TP Rate	FP Rate	Precision	Recall	F-Measure	ROC Area	Class
0.663	0.098	0.616	0.663	0.639	0.917	Wall Crash
0.925	0.048	0.822	0.925	0.871	0.971	Human Crash
0.74	0.036	0.74	0.74	0.74	0.918	Phone Drop
0.958	0.017	0.92	0.958	0.939	0.988	Sharp Turn
0.852	0.011	0.92	0.852	0.885	0.974	Unpaved Road

C. *Random Forest*

Random Forest algorithm performed the best among those tested. As Table 4 shows, like others, the algorithm was not able to identify the Phone Drop very accurately. However, it was fairly successful in identifying the human crash class and the other false positives like sharp turn and unpaved road.

TABLE 4. PERFORMANCE OF THE RANDOM FOREST CLASSIFIER

TP Rate	FP Rate	Precision	Recall	F-Measure	ROC Area	Class
0.7	0.098	0.629	0.7	0.663	0.917	Wall Crash
0.925	0.045	0.831	0.925	0.876	0.975	Human Crash
0.6	0.019	0.811	0.6	0.69	0.926	Phone Drop
0.986	0.015	0.934	0.986	0.959	0.999	Sharp Turn
0.907	0.006	0.961	0.907	0.933	0.998	Unpaved Road

The ROC curve for Random Forest is shown in Figure 5. The Random Forest classifier with the wrapper method for attribute selection was subsequently implemented in the Android application.

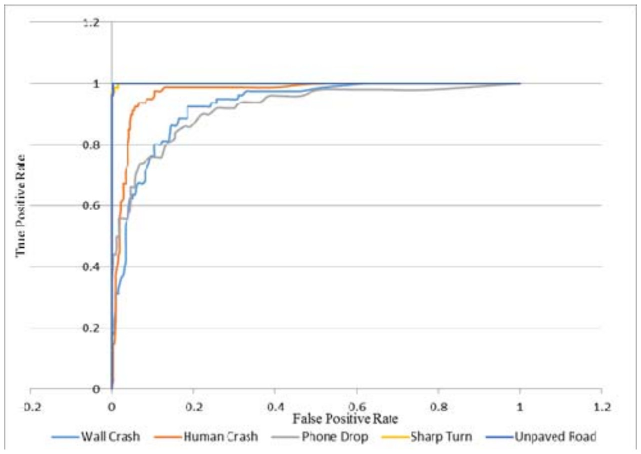
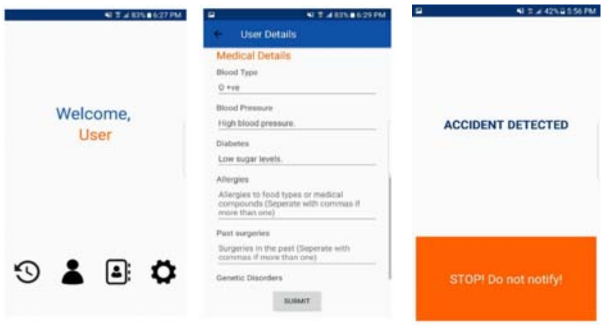


Fig. 5. ROC Curves for Random Forest Classifier

V. Android Application

The android application is called ‘Crash Detect’. The application has been design to ensure ease of use with little or no training. The application currently supports only the English language although in the future, it can be modified to support other languages.


Welcome,
User

User Details
Medical Details
Blood Type
O +ve
Blood Pressure
High blood pressure.
Diabetes
Low sugar levels.
Allergies
Allergies to food types or medical compounds (Seperate with commas if more than one)
Past surgeries
Surgeries in the past (Seperate with commas if more than one)
Genetic Disorders
SUBMIT

ACCIDENT DETECTED
STOP! Do not notify!

Fig. 6. Sample screenshots of the Android application

Upon first run of the application, the user is requested to register his/her details. On this registration page, the user is asked to enter an e-mail address, full name, phone number and password. Personal details requested are so that the Police will be able to identify the user in case of a collision. The user is then asked to enter details regarding medical history and add a number of emergency contacts. The insurance details and date of birth are so that the EMS will be able to directly contact the insurance provider if necessary.

As a service running in the background, the application takes readings of the phone’s accelerometer and gyroscope every 5 milliseconds, feeds this date to the trained model and determines if an accident has taken place. The application automatically sends out a notification to local police, EMS and emergency contacts in case of an accident. This SMS notification contains the type and location of the accident. The user has the option of cancelling the notification in case of a misclassification by the algorithm. Figure 6 show the user interface of the application.

VI. Conclusions

Road accidents are a major cause for concern in the present times. Despite rules and regulations and fines, the death toll and casualty rate caused by road accidents alone are still astounding. There is an urgent need for solutions that increase efficiency of emergency response in these situations. Although car manufacturing companies offer viable solutions with their built-in accident notification systems, these are expensive and could be rendered obsolete with changing technology. This paper presents a solution that is affordable, portable and needs little to no effort on the user’s part. The solution is an Android application that uses an Artificial Intelligence classifier. The application constantly monitors phone sensor data, feeds this data into the classifier model to detect if an accident has occurred. In case of an accident, the application notifies emergency contacts, local police and local emergency medical response teams of the location and classification of the accident.

The model was trained using data collected from laboratory simulations and real cars. Some false positives like Phone Drop were not identified well and need more work. However, this application is a stepping stone towards improving road accident responses in a cost-efficient manner.

References

[1] Hirose et al. “Analysis of Road Accident Rates Following Performed Actions Associated to Engineering, Education and Enforcement: Araraquara, Franca, Matão, Ribeirão Preto and São Carlos,” in Proc. of the 17th *International Conference Road Safety On Five Continents*, 2016.

[2] M. Taamneh, S. Alkheder and S. Taamneh, “Data-mining techniques for traffic accident modeling and prediction in the United Arab Emirates,” *Journal of Transportation Safety & Security*, 146-166, Apr. 2016.

[3] B. Hallinan, “The EMT's and Paramedic's Role in Vehicle Extrication,” *Journal of Emergency Medical Services*, 2015.

[4] D. Diaz, T. Otano, B. Fraile, C. Louis, J. Ramírez, and A. Sucunza, “Use of a Structural Deformity Index as a Predictor of Severity Among Trauma Victims in Motor Vehicle Crashes,” *The Journal of Emergency Medicine on ScienceDirect*, 43(1), 19-28, 2012.

[5] C. Pignataro, “Automatic Crash Notification: A Promising Resource for Fire EMS,” 2013. Available at: http://www.fireengineering.com/articles/print/volume-166/issue-9/departments/fireems/automatic-crash-notification-a-promising-resource-for-fire-ems.html

[6] J. White, C. Thompson, H. Turner, B. Dougherty and D. Schmidt, “WreckWatch: Automatic traffic accident detection and notification with smartphones,” *Mobile Networks and Applications*, 16:285, 2011.

[7] C. Vijayagopalraj, “A Vehicle - Collision Learning System Using Driving Patterns on the Road,” *Masters of Science, University of North Texas*, 2013.

[8] A. Meier, M. Gonter and R. Kruse, “Precrash Classification of Car Accidents for Improved Occupant Safety Systems,” *Procedia Technology*, vol. 15, 198-207, 2014.

[9] F. Aloul, I. Zualkernan, R. Abu-Salma, H. Al-Ali, and M. Al-Merri, “ibump: Smartphone application to detect car accidents,” in Proc. of the *IEEE International Conference on Industrial Automation and Information & Communications Technology*, 52-56, Aug. 2014.

[10] J. Lahn, H. Peter and P. Braun, “Car Crash Detection on Smartphones,” in Proc. of the 2nd *International Workshop on Sensor-based Activity Recognition and Interaction*, 2015.

[11] S. Arivazhagan and L. Ganesan, “Texture classification using wavelet transform,” *Pattern Recognition Letters*, 24(9-10), 1513-1521, June 2003.

[12] D. Li, T. Bissyande, J. Klein and Y. Traon, “Time Series Classification with Discrete Wavelet Transformed Data,” *International Journal of Software Engineering and Knowledge Engineering*, vol. 26, 1361-1377, 2016.

[13] P. Chaovalit, A. Gangopadhyay, G. Karabatis and Z. Chen, “Discrete wavelet transform-based time series analysis and mining,” *ACM Computing Surveys*, 43(2), 1-37, Jan. 2011.

[14] L. Lei, C. Wang and X. Liu, “Discrete Wavelet Transform Decomposition Level Determination Exploiting Sparseness Measurement,” *International Journal of Electrical, Computer, Energetic, Electronic and Communication Engineering*, 7(9), 2013.

[15] A. Seward, “A fast HMM match algorithm for very large vocabulary speech recognition,” *Speech Communication*, 42(2), 191-206, Feb. 2004.

[16] I. Zualkernan, K. Assaleh, S. Dabrai, M. Hoque and H. Pedhiwala, “A Wireless Electronic Training System for Cricket,” in Proc. of the *IEEE 13th Conference on Advanced Learning Technologies*, 2013.

[17] E. Justino, F. Bortolozzi and R. Sabourin, “A comparison of SVM and HMM classifiers in the off-line signature verification,” *Pattern Recognition Letters*, 26(9), 1377-1385, Jul. 2005.