Smartphone based system for real-time aggressive driving detection and marking rash driving-prone areas

Beepa Bose, Joy Dutta, Subhasish Ghosh, Pradip Pramanick and Sarbani Roy
Department of Computer Science and Engineering, Jadavpur University, Kolkata-700032, India
beepabose@gmail.com, joy.dutta.in@ieee.org, subhasish.ghosh1992@gmail.com, pradipcyb@gmail.com and sarbani.roy@ieee.org

ABSTRACT
Integration of the physical world with the computerized world has led to the manifestation of Cyber-Physical Systems (CPSs) in an attempt to build a better and smarter world. In this paper, such a CPS named *D&RSense* has been proposed to promote smart transportation in order to make travelling more comfortable and safe. By studying driving patterns of drivers, *D&RSense* can get valuable insights to their braking and accelerating styles which can help to give them real-time warnings when they drive aggressively. Detection of rash driving prone areas across the city can help to recommend which areas of the city need stricter surveillance. *D&RSense* involves smartphones of commuters and utilizes their accelerometer and GPS sensors to detect driving events like braking and acceleration as well as poor road conditions like bumps and potholes by applying the ensemble learning method for classification, Random Forest (RF). The accuracy of the same has been compared to other supervised machine learning classifiers like Naïve Bayes, k-Nearest Neighbours (k-NN), Decision Trees (DT), Support Vector Machine (SVM) and Artificial Neural Networks (ANN). Rash-driving prone areas and poor road segments during the course of the experiment have been plotted using Density-based spatial clustering of applications with noise (DBSCAN) algorithm. Effectiveness of the proposed application has been evaluated through extensive testing.

CCS Concepts
• **Computing methodologies** → **Machine learning**
• **Human-centered computing** → **Participatory design**

Keywords
Driving event detection; smart transportation; Random Forest; DBSCAN; real-time systems

1. INTRODUCTION
World Health Organization has shown that nearly 1.25 million people die in road traffic accidents each year [1]. In fact, the American Automobile Association conducted a research showing that unsafe driving accounted for more than 56% of road accident fatalities [2]. Now in order to make driving safer, monitoring the traffic system by investing in loop detectors, road-side speed indicators or constant vigilance through CCTVs throughout a city can become pretty expensive. This becomes a vital issue when states lag behind in driver safety improvement programmes and proper road maintenance due to financial crunches along with disregard of drivers toward traffic safety rules. So, a *cost-effective* and *real-time* solution for road safety monitoring was very much in need since smart transportation system is one of the most essential components for realizing a smart city [3].

Workshops ICDCN '18, January 4–7, 2018, Varanasi, India
© 2018 Association for Computing Machinery.
ACM ISBN 978-1-4503-6397-6/18/01 $15.00
https://doi.org/10.1145/3170521.3170549

Smartphones, nowadays, come equipped with a variety of in-built sensors like GPS, accelerometer, magnetometer, gyroscope, proximity sensors, etc. Utilizing the data gathered from these sensors one can detect various activities and events. As a result, smartphone based event or activity recognition has become quite popular and using the smartphone sensors to detect aggressive driving events incur almost no cost at all. One can resourcefully monitor the driving pattern as well as detect road anomalies like bumps and potholes using the information gathered from his or her smartphone. Also, the widespread availability of smartphones presently makes it a very convenient choice for this purpose.

Cyber-physical systems (CPSs) integrate computing and communication with monitoring and control of entities in the physical world. So, a vehicular CPS goes beyond traditional systems with regard to its complexity, requiring specific tools for the analysis of the various properties of transportation.

In this paper, we have proposed an economical and highly efficient CPS named *D&RSense* (Drive and Road Sense) comprising of the smartphones of travellers who contribute their cruise data in this participatory sensing approach in order to make driving safer and more comfortable. Interaction with the physical world i.e. the road and vehicle occurs through the smartphone sensors as well as through the various feedbacks given back to the driver after processing in the system. The proposed system works in a two-fold way:

a) *D&RSense* detects driving events efficiently in real-time and warns the driver immediately when he drives aggressively to avoid accidents. It utilizes the sensors built in Android based smartphones namely the GPS and accelerometer and applies the well-known machine learning approach Random Forest (RF) classifier. The entire processing is done in the smartphone itself.
b) *D&RSense* also utilizes the entire data recorded during different trips from various drivers across the city for further analysis using another RF classifier in a cloud server infrastructure to improve the classifier's learning of event patterns. It also detects poor road segments and rash-driving prone areas in the city.

Now, if a driver, who is trying to continue with his aggressive driving tendencies or is trying to evade the warnings, decides not to use the application, we can still use the passengers' phones to monitor his driving. These data in the long run can also be useful in analyzing the risk profile of drivers which could be utilized by insurance companies for premium adjustment purposes. Moreover, an important area of the system concentrates on understanding the importance of giving real-time alerts to drivers, potentially leading to a very cost effective approach in order to reduce road mishaps.

Now among the common modes of transportation, the bikes are seen to be at the highest risk when it comes to road accidents. The U.S. government's data from 2014 show that for every mile traveled, that year, the number of motorcycle-related deaths was 27 times the number of car-related deaths [4]. So our initial experiments have been concentrated solely on bike transportation data.

D&RSense has been built keeping time as well as accuracy in mind. All our experiments have been carried out on real world data to detect driving and road anomalies in an attempt to build a smart city.

The organization of the paper is as follows. Related works are discussed in Section 2 while the problem statement is presented in Section 3, followed by the proposed approach in Section 4 where the

general process overview and system architecture have been explained in detail. The experiments and results along with implementation are then discussed under Evaluation in Section 5. Finally, the paper concludes in Section 6.

2. RELATED WORKS

Cyber-physical systems for designing a smart transportation system are gaining a growing importance in modern lives and have been a very interesting area of research. In the paper by Wan, Zhang, Zhao, Yang and Lloret, the authors have discussed a Context-Aware Vehicular CPS architecture along with its various challenges and security issues [5]. Moreover, smartphone based event detection in the way of building an intelligent transportation system has been a quite sought after area of research in the recent years. Saiprasert, Pholprasit and Thajchayapong, has shown the detection of driving events using data collected from GPS receiver and accelerometer of smartphones, conveniently without the aid of other external sensors or hardware [6]. Then, Johnson and Trivedi, in their paper [7], have proposed an application called MIROAD, which used classical Dynamic Time Warping (DTW) algorithm on the data gathered from smartphone sensors and showed that the app detected aggressive driving events successfully. Now, DUII (Driving under the Influence of Intoxicants) has always been a major cause of road mishaps. The paper [8] by Dai, Teng, Bai, Shen and Xuan, shows that the proposed program running on Android G1 phones detects drunken driving maneuvers utilizing the data sensed from the phone's accelerometer and orientation sensors with very low percentage of false positive and false negatives.

Using machine learning to process the sensor data for detection of driving events have been shown by Chandrasiri, Nawa and Ishii in their paper [9] where the authors have demonstrated the assessment of driving skills utilizing the data obtained from a driving simulator and then by using the machine learning algorithms, k-NN and SVM on it. Ly, Martin and Trivedi, have put forward a research of driver style recognition and classification utilizing the accelerometer and gyroscope sensors of smartphones along with machine learning algorithms, SVM and k-means clustering. [10] A real-time online prototype driver-fatigue monitor using Bayesian networks were designed by Ji, Zhu and Lan to generate alerts when the driver was detected to be drowsy. [11]

Getting motivated from the above mentioned papers, our proposed system *D&RSense* has been designed which has tried to overcome their limitations as well. *D&RSense* is a participatory sensing approach which overcomes the expenditure issues of infrastructure based monitoring. It consists of a simple and efficient application that runs successfully on Android based smartphones.

The novel contributions of this paper are:

a) All experiments have been conducted on real data gathered from GPS and accelerometer sensors of smartphones while riding bikes.
b) For driving event and road condition detection, a comparison of the machine learning methods, NB, k-NN, ANN, SVM, DT and RF has been performed before drawing inferences.
c) In order to successfully detect the locations where drivers tend to drive aggressively or road segments are damaged, location clustering has been done using DBSCAN.

3. PROBLEM STATEMENT & OBJECTIVES

Let each observation of sensor data get recorded as,

$$\text{sensedData} = \{ lat, long, ts, acc\_x, acc\_y, acc\_z \}$$

where, *lat, long*, *ts*, *acc_x, acc_y* and *acc_z* represent the latitude, longitude, timestamp and magnitude of acceleration from the phone's tri-axial accelerometer along x, y and z axis respectively.

Now let raw tripData = $\{sensedData\}_{ts}$ where, ts = 1,…,T i.e. time at end of trip.

Let an *event e* be a subset of *tripData* such that e = {*sudden_brake, sudden_acceleration, bump, pothole, normal_data*}

Given a *tripData*, we try to detect the set *e* i.e. the series of events that occurred during the course of the trip.

The detailed objectives of *D&RSense* so far can be summed up as:

i. Identify driving events like sudden acceleration, braking and bad road conditions like bumps and potholes from the real travel trajectory information of a biker along with the information gathered from biker's smartphone's tri-axial accelerometer.
ii. Analyze real-time streams of travel trajectory information of bikers and generate immediate warnings when the biker rides aggressively.
iii. Identify rash driving-prone areas and road segments in poor condition.

4. PROPOSED APPROACH

4.1 General Machine Learning Steps Followed

The model is based on pattern recognition of events using supervised machine learning. So the overall steps of machine learning followed in this paper are discussed below.

4.1.1 Data Collection

Raw data i.e. (tripData) D is collected through smartphone sensors while travelling on bikes. Prior to testing, 300 driving templates of varying sizes, corresponding to each of the driving events viz. normal and sudden brake and acceleration as well as road conditions viz. potholes and bumps were prepared to train the classifier. The training set is continuously growing (450+ observations) with time through new template creation as well as through the following steps.

4.1.2 Pre-processing

After data collection, D is essentially mapped to a set of features through the relation,

$$g : D \rightarrow F,$$

where, F is the set of labeled data features.

The raw collected data undergoes steps of pre-processing that includes noise filtration and feature derivation.

a) Filtration of Noise. As shown in Fig 1., the simple moving average (SMA) filter is used for signal smoothing by elimination of noise as much as possible. The main advantage of SMA is that it offers a smoothed line, which is less prone to whipsawing up-n-down because of slight, temporary value swings. Therefore, it provides a more stable level indicating support, which is an essential requirement for building a template in our work.

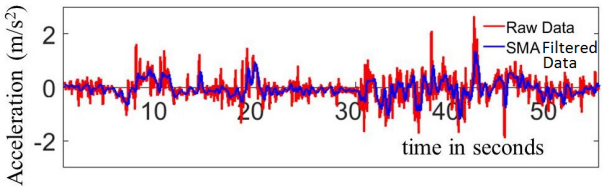

Figure 1. Filtering of raw data using SMA filter.

b) Deriving Features. The raw data has latitude, longitude, timestamp, accelerations along x, y and z axes as features. For each template more features are derived namely, variance (var), standard deviation (sd), maximum of (max), minimum of (min), position of maximum (pmax) and minimum (pmin) acceleration, zero crossing rate (zcr), mean, median (med), interquartile range (iqr), skewness (skew), kurtosis (kurt) for the acceleration along each of the three axes. Along with these, the distance (dist), time duration (tim_dur),

speed and signal magnitude area (sma) for each template are computed. A total 40 features of there were used for this experiment.

Here, distance is calculated [12] as:

$$a = \sin^2(\frac{\Delta\varphi}{2}) + \cos(\varphi 1).\cos(\varphi 2).\sin^2(\frac{\Delta\lambda}{2})$$

$$c = 2.\tan^{-1}(\sqrt{a}, \sqrt{1-a})$$

$$distance = \mathrm{R}.\ \mathrm{c}$$

where, φ, λ, R, are latitude, longitude, earth's radius (mean radius R = 6,371km) respectively.

Thus,

F = {P, L } where P is the set of the derived features used and L are the corresponding labels.

4.1.3 Training

The above features along with the event tags are used to train the machine learning model. In this system, a Random Forest Classifier has been used. One of the most promising ensemble learning algorithms, Random Forest outperforms other supervised learning classifiers like Naïve Bayes, SVM, k-NN, Decision Trees and Artificial Neural networks in terms of accuracy in this case as shown in the result section.

Now the goal of ensemble methods is to combine the predictions of several base estimators' built with a certain learning algorithm in order to improve generalizability or robustness over a single estimator. In this case, the forest has been grown with an ensemble of 500 trees without any pruning and the best split at each node is based on Information gain. 500 was the minimum number of trees selected beyond which there was no significant gain. The number of random features selected for each tree was 9 and finally the categorization result was derived by selecting the label with the highest number of votes from the trees. The features influencing the data, in increasing order of influence were found to be time_dur, var_z, var_y, pmax_x, var_x, iqr_y, mad_y, pmax_y, mean_y, med_y, pmax_z, min_y, max_x, min_z, med_z, kurt_y, zcr_x, max_y, zcr_y, kurt_x, mean_x, mean_z, sma, skew_x, max_z, mad_z, iqr_z, skew_z, min_x, kurt_z, med_x, skew_y, zcr_z, mad_x, and iqr_x.

4.1.4 Event Detection

Since we are processing streaming data and doing real-time event pattern detection, a sliding window of size N and step size N/2 is used where N equals to 5 seconds. By sliding the window to N/2, half of the previous window's data is incorporated in the current window which eliminates the chances of missing patterns from the filtered data as shown in Fig.2.

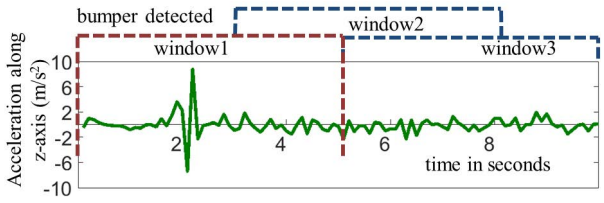

Figure 2. Event detection on filtered data that captured the road bump effectively.

4.1.5 Classification

Once the test data are classified using RFC, the data corresponding to the detected events can be added to the training set along with the predicted labels that will make the training dataset ever increasing. Thus,

L → {sudden_brake, normal_brake, sudden_acceleration, normal_acceleration, bump, pothole}

4.2 Architecture

The architecture of the system of *D&RSense* has been represented in Fig. 3, which consists of two layers, namely the *Crowd Layer* and the *Cloud Layer*.

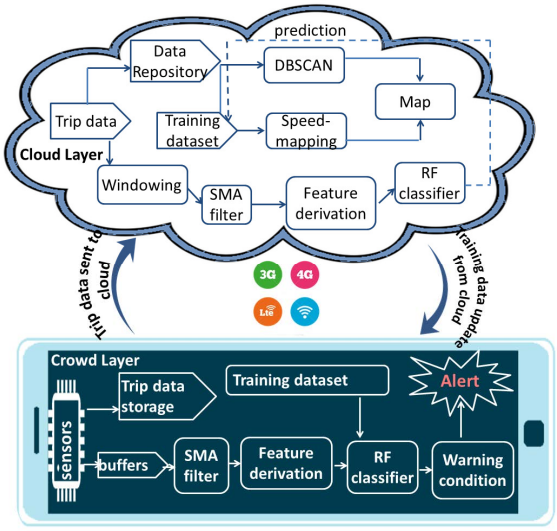

Figure 3. Architecture of D&RSense.

4.2.1 Crowd Layer

The bottom layer of the model is the Crowd Layer. It consists of the smartphones of drivers and passengers in the city that have the *D&RSense* application installed and running. In this phase, the model utilizes the incoming real-time stream of transportation data from GPS and accelerometer of smartphones and instantly detects the driving anomalies, i.e. aggressive driving events by using a Random Forest Classifier (RFC) and aims at giving immediate warnings to the driver. This detection does not require any active Internet connection and is done locally in the respective smartphones.

Table 1. Computational complexities of discussed classifiers

Classifier	Build/Training Complexity	Prediction Complexity
RF [13]	$O(M*F*N^2 \log N)$	$\Theta(M*N)$
NB [14]	$O(N*F)$	$O(C*F)$
SVM [15]	$O(\max(N, F) \min(N,F)^2)$	$O(N_{sv}*F)$
k-NN [16]	O(k*N*F) Here, O (N*F) as k=1	
DT [13]	$\Theta(F*N^2)$	$\Theta(N)$

M: number of trees, N: number of observations, F: number of features, C: number of classes, N_{sv}: number of support vectors.

Now this phase is essentially aimed at generating real-time alerts as well as ensuring accuracy. So, there has to be a trade-off between the two as usually greater accuracy is achieved with a higher computational complexity. Table 1 shows the various computational complexities of the discussed algorithms in this paper. In case of k-NN, choosing an optimal value of k and proper feature selection becomes crucial. Moreover k-NN is a lazy learner i.e. instead of finding a discriminating function from the training data, it memorizes the entire training dataset thereby making the prediction step very expensive. So, it will not perform well when the amount of data will become huge.

Now data picked up by smartphone sensors will have a lot of noise and outliers. Both k-NN and SVM are sensitive to that. Moreover, SVM is intrinsically suited for two–class problems while ours is a 6-class problem here. Again, Naïve Bayes makes the very strong assumption that the features are conditionally independent given the class, which is not always true. It is extremely fragile to over-fitting when new data comes in, just like Artificial Neural Networks and

Decision Trees. Traditional ANN also fails to handle missing values and suffers sometimes because of local minima.

Now a Random Forest is an ensemble model represented by a set of unpruned decision trees, grown, based on multiple bootstrap samples drawn with replacement from the training set, with randomized split selection. Though computationally more expensive than DTs, RFs are much more robust. The Decision Tree classifiers uses greedy approach and hence an attribute chosen at first step cannot be used anymore which can give better classification results if used in later steps. Also it can over-fit the training data.

A major advantage of using Random Forests is that the over-fitting problem will never happen when one uses the algorithm in any classification problem. So the classifiers running locally will never get biased and also there is no need for normalizing or scaling the data or imputing missing values before using RFC. Random forests can run on very large datasets where the feature space is huge without deletion of any variable. Also, near-zero variance features are ignored. So, after testing with NB, k-NN, ANN, SVM, DT and RF, RF was used for pattern matching and event detection for quick alert generation in this phase. The steps are:

a) Data Collection. From the moment the user activates the "Driving" mode of the application, data collection begins; raw time-stamped data from the phone's accelerometer and GPS starts getting recorded within the smartphone itself.

b) Event Recording. Local data processing begins whenever the sensors' signal reading crosses the thresholds as shown in Table 2 and explained through the algorithm following step *d*. A buffer is maintained where we extract the data preceding and succeeding for 2.5 seconds each, since the occurrence of the threshold. We use this 5 seconds of data for local event detection which is enough to capture any event for local processing.

Table 2. Accelerometer Thresholds [6]

Event	Axis	Threshold set
Sudden acceleration	x-axis y-axis	> = |0.3| G 2.5 G > y > 0G
Sudden braking	x-axis y-axis	> = |0.3| G -2.5G < y <= 0G
Bump	x-axis z-axis	>= |1|G >= |6|G
Pothole	x-axis z-axis	<= -2G >= |6|G

c) Event Detection. The 5 seconds data collected for local processing is analyzed using a Random Forest classifier after pre-processing as discussed before, and then classified accordingly.

d) Alert. If within a time span of 5 minutes, the event identified by the RF classifier is the third or higher consecutive occurrence of any of the sudden acceleration or sudden braking event in number, an instant warning is generated stating the anomaly. However, if it is seen that the biker was within the specified speed limit for the road and the road condition is bad for more than 5 seconds, then the aggressive driving is excused and no alert is generated.

The overall algorithm running in the smartphones during this phase is given next:

Algorithm:

Init()

```
{
        tripData tD= NULL
        boolean tripON= FALSE
        event[] buffer = NULL
        event eventDetected = NULL
        String roadCondition = "GOOD"
}
```

Main()

```
Init();
Login(); /*driver logs in*/
tripON = TRUE;
while (tripON == TRUE)
{
    tD = dataCollection();
    while(checkEventPossibility (tD) == TRUE)

/*checkEventPossibility: if (|acc_y| >= 3.5 && |acc_x| <2.5) at any
point in tD*/

    {
        buffer = extractData(tD, tStart,tEnd);

/*extractData() : extracts the 2.5 seconds data preceding and 2.5
seconds succeeding the point where the threshold condition
matched, thus for 5 seconds*/

        eventDetected = randomForestClassifier (buffer);
      if(eventDetected == "SuddenAcceleration" ||
eventDetected == "SuddenBrake")
        {
            if((checkDriving() == "AGGRESSIVE" &&
roadCondition != "BAD") || speed > limit)

/* if this is the 3rd or higher occurrence of an aggressive driving
event within a span of 5 minutes while road condition is not poor or
if speed is higher than the said limit */

                {
                    generate WARNING
                }
        }
    }
    tripON = checkTripStatus()

/*if driving has been turned off exit while*/

}
sendToServer(tD)
```

Now, there will be a fixed upper limit in the volume of recorded event pattern data that can be stored in the smartphone as the training dataset. However, its contents can still be modified periodically with updates from the cloud. Once the trip is over, the "Driving" mode is turned off in the *D&RSense* app and all the raw data recorded throughout the trip is sent to the cloud server over the Internet for further processing.

4.2.2 Cloud Layer

The volume of data collected from numerous drivers and passengers in the city in this system over various trips on different days will gradually become too huge to be running on a single smartphone. Moreover different drivers across the city will have different patterns of generating aggressive events. To build a strong detector, one needs to train the classifier using all the different patterns. So in order to utilize the different patterns of the driving events generated by different drivers, to detect the damaged road segments as well as locations where drivers tend to drive aggressively in the city and to meet the need of extra storage and further processing of the huge data we have the Cloud Layer as shown in Fig. 3. The Cloud has a data repository where all the raw data from various trips are kept. It also has another Random Forest Classifier (RFC) that runs on the data gathered from different drivers in the city. Following the steps discussed in the general overview section, event classification was performed using the RFC and once classified, the predicted data along with the labels can be added to the training dataset which keeps on growing and updates from it are sent to the training datasets in smartphones every week over the Internet.

We can also cluster the aggressive events occurring at different times of the day across the city by using DBSCAN algorithm as it works bottom-up by picking a point and looking for more points within a given distance. It then expands the cluster by repeating this process for new points until the cluster cannot be expanded further. K-means, though simple, will not be suitable for this purpose as we cannot know the value of k beforehand. Moreover in case of k-means, every co-ordinate must belong to a cluster. Hierarchical clustering is also not suitable for very large datasets making it unsuitable for our problem.

Areas where driver crosses speed limits can be inferred as well, by utilizing the GPS data and a reverse geo-mapping of speed limits can be done as shown in the result section.

4.3 Implementation

Android Studio 2.3.1 was used as IDE for both design and development of the application. In our work, we have used OPENSHIFT [1] as our Cloud Service provider. The background processing in the cloud is done in R Programming language.

5. EVALUATION

5.1 Experimental Setup

The experiment was carried out using the 5 different Android based smartphones, Redmi 3S Prime (3GB), Redmi Note 4 (2 GB), Samsung Galaxy A7 (3GB), Motorola Moto E3 (1GB) and Samsung GT-S7392 (512 MB) each of which was equipped with GPS, accelerometer sensors and had the *D&RSense* application installed. We have carried out the experiment on data collected during different bike trips. The 2 different bikes used in this experiment are one Bajaj Avenger Cruise (220Cc Model) and one Bajaj Vikrant (V15) (150cc Model) which were ridden by two different drivers.

The dataset was prepared by collecting bike data along the route from Jadavpur University Main Campus to Baruipurvia Baruipur Bypass[2] over an approximate distance of 50 km (minimum) every day from 28th July 2017- 15th September 2017 over a period of 50 days. Various driving events like normal and sudden acceleration and braking, bumps and potholes were successfully recorded. Both types of acceleration templates (normal and sudden) were prepared while increasing the speed from 0-20, 0-40, 0-60, 20-40, 20-60 and 40-60 kmph and both the types of the braking templates was made while decreasing the speed in reverse of all the previous mentioned values (for normal and sudden acceleration). After rigorous data collection phase by the volunteers, we got the initial dataset for the training and testing purpose consisting of 300 observations which have increased to over 450 during the course of the experiment. For the local phase, the size of the training dataset was fixed and was updated every 7 days while in case of the cloud, the training dataset increased continuously over the 50 days of the experiment which in turn, improved the system's accuracy and when the results seemed to be more or less stable, then only inferences were drawn. The smartphone was held horizontally orienting its axes along with the bike axes during the data collection.

For testing of long datasets, both the drivers, each with a different bike drove over the mentioned route sometimes calmly and sometimes aggressively. From this data, the event templates were prepared in such a way that the initial training data had an equal number of the specific events (normal as well as anomalous).

5.2 Results

After setting all the conditions as mentioned above, both the ends, i.e. local and cloud part of the system were tested for the system's accuracy validation. We also tested with a Naïve Bayes classifier (after a feature extraction step using Random Forest which worked better than PCA in this case), a k-NN classifier, an ANN classifier, an SVM classifier (while using PCA for feature extraction in these 3 cases) and a Decision Tree classifier. While testing the accuracies as mentioned before, RF gave the highest accuracy of 94.07%, followed by NB, SVM, ANN, k-NN and DT that gave accuracies of 89.57%, 86.16%, 86.147%, 85.18% and 84.74% respectively.

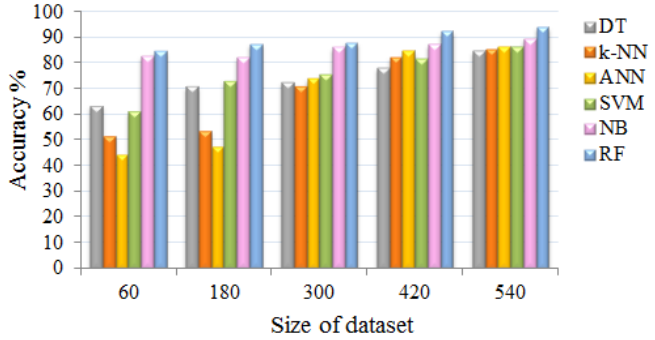

Figure 4. Accuracy of the techniques improves as the size of training dataset increases.

Fig. 4 shows that accuracies of all the 6 classifiers improve as the size of the training dataset increases. Also, it actually gives the system's overall accuracy, not fine grained one for individual events.

4/5th of the dataset was used to train the model while using a repeated 5-fold cross-validation and remaining 1/5th was used to do the testing. Each accuracy was defined as follows:

$$Accuracy = \frac{C_e}{T_e} \times 100\ \%$$

where, C_e: Total number of correctly classified events.
T_e: Total number of events.

Now, when looking into the details of each dataset, the correct predictions along with the false positives and false negatives for events like sudden acceleration, sudden brake, bumper and pothole which affect driving directly leading to accidents need to be analyzed separately and highlighted to understand the system's performance closely. Moreover it will directly affect the real-time alerts generated during the local phase. So, Fig. 5 highlights correctly detected, false positive and false negative events for the RFC based on the total dataset available for the training and testing purposes.

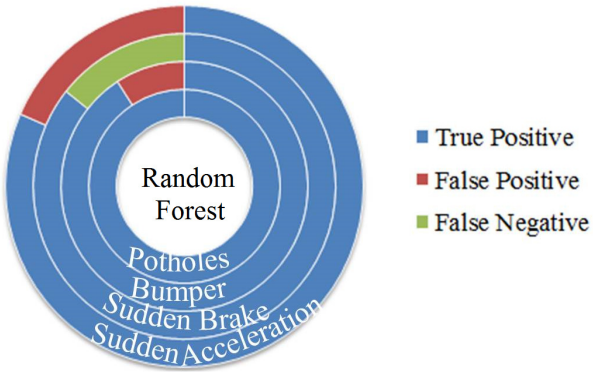

Figure 5. Event detection by RFC.

Now the speed module of *D&RSense* not only acts as a real-time speedometer for warning purposes by giving alert to the driver when over-speeding but also helps to maintain locations in the cloud where over-speeding tends to occur. According to the Traffic Regulations of Kolkata Traffic Police, the speed limit for riding over the A.J.C. Bose Road flyover on a two-wheeler is 50kmph [17]. So for testing purposes as we rode over the flyover, at a speed of 60-70 kmph *D&RSense* kept issuing alerts as shown in Fig 6 from screenshots of the app. It displays a portion of the map that can be provided to the authority which clearly displays where over-speeding had occurred.

[1] OPENSHIFT
[2] EM Bypass, Kolkata, West Bengal, India

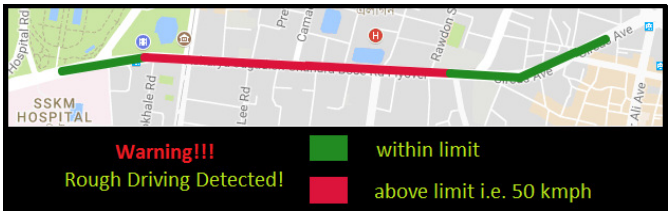

Figure 6. Part of apllication's screenshot in smartphone displaying alert generated when biker crossed speed-limit.

Now, Fig 7 represents the various events detected by the local phase RFC in order to alert the biker when driving aggressively. However one can see it excuses scenarios where the biker braked suddenly due to the presence of bumps or potholes while staying in speed limit.

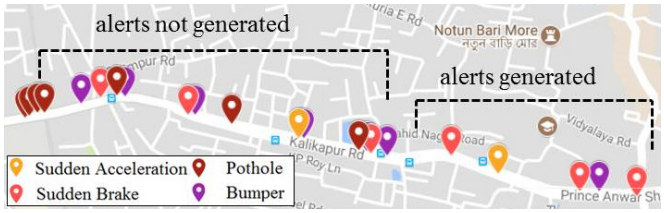

Figure 7. The various events detected by RF mentioning where alerts were generated.

The various trip data collected were classified in the Cloud as discussed in the Architecture section before in order to locate the various events detected across the trips so that we can infer the rash-driving prone areas within the experimental locations. Fig 8(a) displays five such areas. DBSCAN clustering was used to cluster some GPS points corresponding to those locations where more than 10 aggressive driving events kept on occurring over a course of 5 days of trip data collection during various times of the day. This information can be very much useful in deciding where traffic in the city will require more careful monitoring. Fig 8(b) shows a road segment marked pothole-filled. *D&RSense* clustered the various potholes detected on the particular road using DBSCAN and marked the segment to be a poor one.

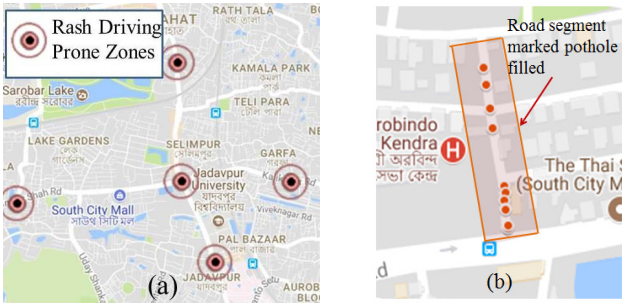

Figure 8. (a)Aggressive event prone areas in Jadavpur area and (b) Road filled with potholes clustered using DBSCAN.

6. CONCLUSION AND FUTURE WORK

In this paper, a CPS called *D&RSense* has been proposed in order to reduce the number of road mishaps and to make transportation more comfortable and safe with the help of real time detection of rash-driving and by identifying poor road conditions in the city. *D&RSense* recognizes aggressive driving events and poor road conditions by studying the readings of accelerometer along with GPS of smartphones. A Random Forest Classifier has been used to do a real-time and aggressive event recognition so that warnings can be issued to the driver timely to prevent road mishaps. The same machine learning technique RF has been applied while processing in the cloud in order to do further analysis and to keep on modifying the locally running classifier's local training dataset. Rash-driving prone areas and road segments filled with bumps and potholes have been marked using DBSCAN in the cloud.

The data collections are still going on and in a few months the system would perform a lot better than as of now when we will have more data. As our future work, we would extend the experiment by addressing challenges like:

i. building different datasets for bike, auto, car and bus at different times of the day successfully through crowd-sensing.
ii. detecting and considering traffic condition or congestion from the sensor data
iii. sharing of travel trajectory information has privacy and security challenges which are yet to be addressed
iv. the experiments that have been conducted so far had the phone held horizontally along the bike. We need to consider virtual reorientation of the phone when placed in different positions.

7. REFERENCES

[1] World Health Organization: Global status report on road safety, http://www.who.int/violence_injury_prevention/road_safety_status/2015/en/

[2] American Automobile Association: Aggressive Driving: Research Update, 2009, https://www.aaafoundation.org/sites/default/files/AggressiveDrivingResearchUpdate2009.pdf

[3] Dutta, J., & Roy, S. (2017, January). IoT-fog-cloud based architecture for smart city: Prototype of a smart building. In *Cloud Computing, Data Science & Engineering-Confluence, 2017 7th International Conference on* (pp. 237-242). IEEE.

[4] National Center for Statistics and Analysis. (2016, June). Motorcycles: 2014 data. (Traffic Safety Facts. Report No. DOT HS 812 292). Washington, DC: National Highway Traffic Safety Administration.

[5] Wan, J., Zhang, D., Zhao, S., Yang, L., & Lloret, J. (2014). Context-aware vehicular cyber-physical systems with cloud support: architecture, challenges, and solutions. *IEEE Communications Magazine*, *52*(8), 106-113.

[6] Saiprasert, C., Pholprasit, T., & Thajchayapong, S. (2017). Detection of driving events using sensory data on smartphone. *International Journal of Intelligent Transportation Systems Research*, *15*(1), 17-28.

[7] Johnson, D. A., & Trivedi, M. M. (2011, October). Driving style recognition using a smartphone as a sensor platform. In *Intelligent Transportation Systems (ITSC), 2011 14th International IEEE Conference on* (pp. 1609-1615). IEEE.

[8] Dai, J., Teng, J., Bai, X., Shen, Z., & Xuan, D. (2010, March). Mobile phone based drunk driving detection. In *Pervasive Computing Technologies for Healthcare (PervasiveHealth), 2010 4th International Conference on-NO PERMISSIONS* (pp. 1-8). IEEE.

[9] Chandrasiri, N. P., Nawa, K., & Ishii, A. (2016). Driving skill classification in curve driving scenes using machine learning. *Journal of Modern Transportation*, *24*(3), 196-206.

[10] Van Ly, M., Martin, S., & Trivedi, M. M. (2013, June). Driver classification and driving style recognition using inertial sensors. In *Intelligent Vehicles Symposium (IV), 2013 IEEE* (pp. 1040-1045). IEEE.

[11] Ji, Q., Zhu, Z., & Lan, P. (2004). Real-time nonintrusive monitoring and prediction of driver fatigue. *IEEE transactions on vehicular technology*, *53*(4), 1052-1068.

[12] Montavont, J., & Noel, T. (2006, June). IEEE 802.11 handovers assisted by GPS information. In *Wireless and Mobile Computing, Networking and Communications, 2006.(WiMob'2006). IEEE International Conference on* (pp. 166-172). IEEE.

[13] Louppe, G. (2014). Understanding random forests: From theory to practice. *arXiv preprint arXiv:1407.7502*.

[14] Fleizach, C., & Fukushima, S. (1998). A naive Bayes classifier on 1998 KDD Cup.

[15] Chapelle, O. (2007). Training a support vector machine in the primal. *Neural computation*, *19*(5), 1155-1178.

[16] Manning, C. D., Raghavan, P., & Schütze, H. (April, 2009): An Introduction to Information Retrieval.,299

[17] https://kolkatatrafficpolice.net/Uploadedfiles/TR_SpeedLimit.pdf