A Novel Two-Step Fall Detection Method Using Smartphone Sensors

John C. Dogan
Computer Science Department
Southern Connecticut State University
New Haven, CT, USA
DoganU1@SouthernCT.edu

Md Shafaeat Hossain
Computer Science Department
Southern Connecticut State University
New Haven, CT, USA
HossainM3@SouthernCT.edu

***Abstract*—A smartphone-based fall detection system has two major advantages over a traditional fall detection system that comes as a separate device: (1) the phone can automatically send messages to or call the emergency contact person when a fall is detected and (2) a user does not need to carry an extra device. This paper presents a novel two-step fall detection method which uses data extracted from smartphone sensors to detect falls. A fall can happen in many ways. A person can fall while he/she is walking, jogging, sitting, or even sleeping. Patterns of all falls are not the same. It is important to identify the type of falls to precisely distinguish it from non-falls (normal activities). Hence, our method first identifies the correct type of falls by performing multi-class classification. In the second step, this method produces a binary decision based on the multi-class prediction. We collected data from 10 users to evaluate our proposed fall detection method. Each user performed five normal activities–namely, walking, jogging, standing, sitting, lying, and also fell after performing each activity. We performed experiments with five common smartphone sensors: accelerometer, gyroscope, magnetometer, gravity, and linear acceleration. We tested five machine learning classifiers–namely, Support Vector Machine, K-Nearest Neighbor, Decision Tree, Random Forest, and Naive Bayes. Our two-step fall detection method achieved the maximum accuracy of 95.65% and the maximum area under ROC curve (AUC) of 0.93, both with the gyroscope sensor and Support Vector Machine classifier.**

***Index Terms*—Fall detection, Smartphone, Sensor, Gyroscope, Accelerometer, Activity recognition.**

I. Introduction

Falls are a major cause of injuries and hospital admissions among elderly people. The estimated fall incidence for both institutionalized and independently living people over 75 is at least 30% every year [1]. According to the Centers for Disease Control and Prevention, each year 2.8 million older people are treated in emergency departments for falls [2]. One out of five such falls causes a serious injury such as broken bones or head injury [3]. The consequences of a fall significantly depend on the time interval during which the person remains unaided after the fall. The conventional solution to the fall detection problem consists of providing people with a Personal Emergency Response System (PERS)–a small, lightweight and battery powered device with a "help" button that can be carried by the elderly. However, the problem is it needs the user to press the "help" button, which may not be possible if the victim loses consciousness. Hence, it is necessary to build a system which can automatically detect falls and send request for help.

According to Ericsson mobility report, global smartphone users are to reach 6.1 billion by 2020 [4]. This means that almost everyone will carry a smartphone and this includes a large population of the elderly. If the smartphones contain the fall detection system, the elderly people do not need to buy and carry an extra device like PERS. Smartphones can automatically send messages to the emergency contact person when a fall is detected.

Most smartphones have several built-in sensors that measure motion, orientation, and various environmental conditions. These sensors are capable of providing raw data with high precision and accuracy, and are useful to monitor three-dimensional device movement or positioning, or to monitor changes in the ambient environment near a device [5]. Since smartphones have become essential in people's lives, a sensor-based fall detection system will make them perfect for assisting life in healthcare [6].

Several recent studies have focused on smartphone-based fall detection systems to reduce the death rate from falls [7], [8], [9], [10]. They came up with several machine learning approaches to detect falls using smartphone sensors. In this paper, we work on improving the performance of smartphone-based fall detection systems. Below, we briefly describe our contributions:

- We develop a new two-step fall detection method which precisely detects falls. Existing fall detection methods perform binary classification and produce a fall or non-fall decision. In contrast, our method first performs multi-class classification and then converts the multi-class prediction to a binary decision. These two steps are crucial for precisely detecting falls. A fall can happen in many ways. A person can fall while he/she is walking, jogging, sitting, or even sleeping. Patterns of all falls are not the same. Hence, our method first identifies the correct type of falls or non-falls (normal activities). In the second step, it produces a binary decision.
- We collected data from 10 users to evaluate our proposed fall detection method. Each user performed five normal activities: walking, jogging, standing, sitting, lying, and also fell after performing each activity. From a raw

walking or jogging sample, where a participant had to walk or jog for 20 seconds, 24 samples of normal activity and 1 sample of fall were created. From a raw standing, sitting, or lying sample, where a participant had to keep standing, sitting or lying for 10 seconds, 12 samples of normal activity and 1 sample of fall were created.

- We performed experiments with five smartphone sensors, namely, accelerometer, gyroscope, magnetometer, gravity, and linear acceleration. This comparative study suggests that gyroscope is the most effective sensor for fall detection in the smartphone.
- We defined 11 new features and combined them with 4 features in existing literature. After extracting 15 features from each axis data, we combined all 45 features of three axes for training and testing our machine learning classifiers.
- We performed experiments with five machine learning classifiers, namely, Support Vector Machine, K-Nearest Neighbor, Decision Tree, Random Forest, and Naive Bayes. This comparative study reveals that the Support Vector Machine is the most accurate classifier for fall detection in the smartphone.
- Our two-step fall detection method achieved the maximum accuracy of 95.65% and the maximum area under ROC curve (AUC) of 0.93, both with the gyroscope sensor and Support Vector Machine classifier.

The rest of the paper is organized as follows. In Section II, we discuss related literature. In Section III, we explain our two-step fall detection method. In Section IV, we describe our data collection app and procedure. In Section V, we describe our preprocessing steps and features we extracted. In Section VI, we describe our experiments and analyze the results. Finally, we conclude in Section VII.

II. Related Research

We divide our discussion on related studies into two subsections: activity recognition and fall detection.

A. *Studies on Activity Recognition*

Human activity recognition has gained a lot of attention in past years due to its high impact in various real world applications. These studies have mentioned that an automatic activity recognition system can be built based on intelligent processing of multiple sensor data on smartphones, to contribute to the eHealth area [6], [11]. Anguita *et al.* [12] demonstrated appealing use of human activity recognition in healthcare applications and developed a human activity recognition system using smartphone inertial sensors. They performed multi-class classification using Support Vector Machine (SVM). Bayat *et al.* [13] used accelerometer data from smartphones to recognize several human activities. They designed a low pass filter to separate the gravity component from body acceleration and after performing experiments with several classifiers, they obtained an accuracy of 91.15%. Deng *et al.* [14] also used accelerometer to recognize human activities. To achieve a better performance, they proposed a cross-person activity recognition model which used reduced kernel extreme learning machine.

Recently, deep learning based approaches have been widely adopted for the sensor-based human activity recognition [15]. Ronao and Cho [16] developed a deep convolutional neural network based human acitivity recognition system using smartphone sensors. Using raw sensor data, they achieved an overall accuracy of 94.79%. Lee *et al.* [17] developed a one-dimensional convolutional neural network based human activity recognition system. Using accelerometer data they achieved an accuracy of 92.71% to recognize three activities: walking, running, and staying still. Inoue *et al.* [18] used deep recurrent neural network (DRNN) to recognize human activities utilizing accelerometer data. After tuning several parameters of the DRNN, they achieved a maximum recognition rate of 95.42%.

B. *Studies on Fall Detection*

Fall detection is not a new research area. For decades researchers have been working to develop a reliable fall detection system. Early developments such as [19], [20], [21], [22], and [23] used accelceromerter and/or gyroscope sensors embedded in some wearable devices to detect falls. The weakness of these systems is the person has to wear an extra device. No doubt, it incurs extra cost in terms of money and user inconvenience.

Several studies have continued this research to build a more accurate and reliable fall detection system. Lim *et al.* [24] uses a Hidden Markov Model (HMM) with 3-axis acceleration to detect falls. Yazar *et al.* [25] uses vibration sensors that converts vibrations into electrical signals depending on the intensity of the vibration waves in the axis of the vibration sensor. Nadee and Chamnongthai [26] uses ultrasonic technology-based multi sensors, which are connected to an Arduino microcontroller in order to send the elderly person's fall related signal using WiFi to the processing unit. The signal is analyzed to recognize human by sensing distance, and detect the action such as standing, sitting, and falling by pattern matching.

With the advent of smartphones, fall detection research received new dimension. Dai et al. [7] is the pioneering work to apply smartphone sensors for fall detection. They developed a fall detection system called "PerfallD" which used accelerometer for fall detection and demonstrated better performance than existing commercial fall detection products. In addition to strong detection performance, they achieved notable power efficiency. This early research done on smartphones paved a path for future research.

Huynh *et al.* [10] studied both accelerometer and gyroscope for fall detection. They provided a systematic approach to selection of critical thresholds for fall detection that simultaneously optimizes both sensitivity and specificity. Hsu *et al.* [9] focused on extracting gravity related features. Specifically, they extracted the acceleration in the gravity vector direction, which contains the vertical directional information and provides a distinct pattern of fall-related activity. Qu *et al.* [27] developed a low complexity algorithm to detect falls using smartphone sensors. Lee *et al.* [8] developed a real time fall detection system using accelerometer sensor of smartphones.

This system conducts real time location tracking utilizing Google's Map and 3D information to cope with a critical situation and to make an urgent intervention in an emergency.

There are some recent studies such as [28] and [29], which combined sensor information from smartphones and smartwatches to create an automatic fall detection system. Because both smartphones and smartwatches have accelerometers and gyroscopes, it was possible to fuse sensor information from both devices. Zhen *et al.* [30] developed a fall detector combining MEMS inertial sensor and smartphone.

Our study puts its complete focus on smartphone to reduce dependency on multiple devices. We perform a comparative study of five smartphone sensors with five machine learning classifiers. The main contribution of this study lies in the novel fall detection strategy which creates profiles of several kinds of normal activities and falls and makes a decision in two steps.

III. Our Fall Detection Methodology

Our fall detection methodology includes two phases: (1) training and (2) testing. A fall can happen in many different ways such as a person can fall from being walking, he/she can fall from being sitting in a chair, he/she can fall from being lying on a bed, he/she can fall while he/she is jogging, etc. Current fall detection studies did not consider these different types of falls. Here is the uniqueness of our study, we deeply focus on each type of falls to perceive their accurate patterns. In the training phase, we create profiles of five normal activities: standing, sitting, walking, jogging, and sleeping/lying and profiles of five falls: fall from standing, fall from sitting, fall from walking, fall from jogging, and fall from sleeping/lying.

Figure 1 shows how our normal activities and falls are profiled. Our collected activity samples include both normal activities and falls. For example, during data collection, we instructed the participant to walk for 20 seconds, then he/she would hear a beep and would fall immediately. We discuss more about our data collection procedure in Section IV. From each activity sample, we separate the normal activity and fall, extract their features, and create the profiles.

In the testing phase, we separate the normal activities and falls from the test activity samples, extract their features, and match against the profiles created in the training phase. Our fall detection (testing) phase involves two steps. First, we perform multi-class classification which specifically tells us what type of normal activity or fall the test sample belongs to. Second, we make a binary: fall or non-fall decision from the output generated in the first step, *i.e,*, we convert the multi-class decision to a binary decision. Specifically, any type of normal activity (output from the first step) is classified as non-fall and any type of fall (output from the first step) is classified as a fall. This two-step fall detection strategy is more precise than the traditional one-step fall detection strategy which does not discriminate different types of normal activities and different types of falls in training and just performs a binary classification.

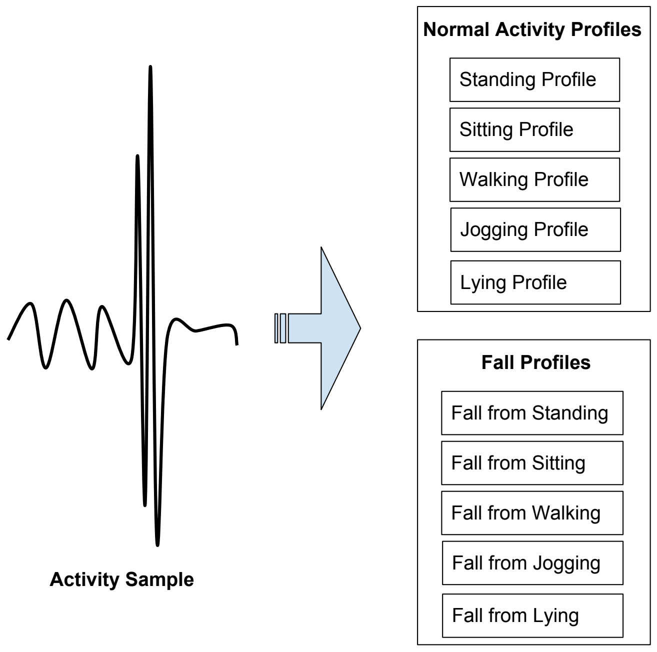


Fig. 1: Creating normal activity profiles and fall profiles from activity samples.

We perform experiments with five smartphone sensors– namely, accelerometer, gyroscope, magnetometer, gravity, and linear acceleration. We define 11 new features and combine them with 4 features in existing literature. After extracting 15 features from each axis data, we combine all 45 features of three axes for training and testing our machine learning classifiers. We perform experiments with five machine learning classifiers: Support Vector Machine, K-Nearest Neighbor, Decision Tree, Random Forest, and Naive Bayes.

IV. Data Collection

We collected normal activity and fall data from 10 users. We created an Android application for data collection. The LG Nexus 5 phones were used to collect data from five sensors: accelerometer, gyroscope, magnetometer, linear acceleration, and gravity. Android sensor data are obtained along three axes: X, Y, and Z. Figure 2 shows the coordinate system used by Android Sensor API.

Accelerometer, gyroscope, and magnetometer are called the base sensors because they collect data from a single physical sensor. An accelerometer sensor collects the acceleration of the device along the 3 sensor axes. A gyroscope sensor collects the rate of rotation of the device around the 3 sensor axes, were the rotation is positive in the counterclockwise direction. A magnetometer sensor collects the ambient magnetic field, as measured along the 3 sensor axes. Linear acceleration and gravity are composite sensors. A composite sensor generates data by processing and/or fusing data from multiple physical sensors. The underlying physical sensors of linear acceleration are accelerometer and gyroscope/magnetometer.

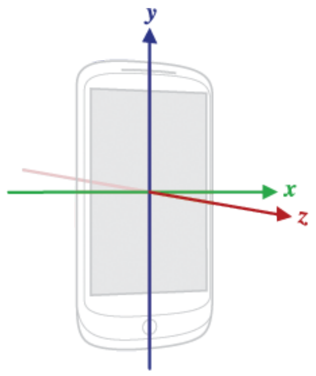


Fig. 2: Three axes of an Android device used by the Sensor API. Source: https://source.android.com/.

The underlying physical sensors of gravity are accelerometer and gyroscope. A linear acceleration sensor collects the linear acceleration of the device in the sensor frame, not including gravity. A gravity sensor collects the direction and magnitude of gravity in the device's coordinates. A detailed explanation of all these sensors is available at: https://source.android.com/devices/sensors/sensor-types.

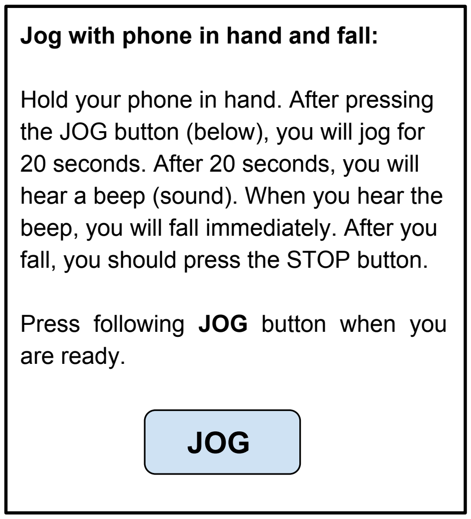
Our data collection application has five modules: stand, sit, walk, jog, and lie. Each module collects the associated normal activity and fall data. Users held the phone in their hand while giving the data. For example, while using the 'stand' module, the participant has to keep standing for 10 seconds with the phone in hand and then fall. The participant does not need to count 10 seconds. After 10 seconds of standing activity, the application produces a beep sound which tells the participant that it is time to fall. So, after standing for 10 seconds, the participant hears the beep and falls immediately. Similarly, while using the 'sit' and and 'lie' modules, the participant has to keep sitting and lying, respectively, for 10 seconds, then hears a beep, and falls immediately. While using the 'walk' and 'jog' modules, the participant has to walk and jog, respectively, for 20 seconds, then hears a beep, and falls immediately. After a fall, the participant has to press a 'STOP' button provided by the application to stop the Android sensors from recording more data. Figure 3 demonstrates the data collection instructions used in 'jog' module of our Android application.

Data were collected in three sessions. That is, a participant performed each activity one to three times. After data collection, we plotted the raw data from each sensor along the three axes. After examining the plots, we found that falls are obvious in the data graphs. Figure 4 demonstrates how falls create a sudden change in the data graph generated from accelerometer sensor data.

V. Preprocessing and Feature Extraction

Each of our collected activity samples consists of a normal activity (such as jog, walk, stand, etc.) followed by a fall. Hence, we split each raw activity sample into two parts: a normal activity part (such as jog part, walk part, stand part, etc.) and a fall part. We generated multiple samples from the normal activity part and one sample from the fall part. Specifically, we extracted multiple windows using 50% overlapping where each window had a size of 1.5 seconds. Each window had been considered as a sample during training and testing the machine learning classifiers. From a raw walking or jogging sample, where a participant had to walk or jog for 20 seconds, 24 samples of normal activity and 1 sample of fall were created. From a raw standing, sitting, or lying sample, where a participant had to keep standing, sitting or lying for 10 seconds, 12 samples of normal activity and 1 sample of fall were created. We extracted features from each of these samples.

Before we extracted features, we cleaned our data. When


Jog with phone in hand and fall:

Hold your phone in hand. After pressing the JOG button (below), you will jog for 20 seconds. After 20 seconds, you will hear a beep (sound). When you hear the beep, you will fall immediately. After you fall, you should press the STOP button.

Press following **JOG** button when you are ready.

Fig. 3: Data collection instructions used in 'jog' module of our Android application.

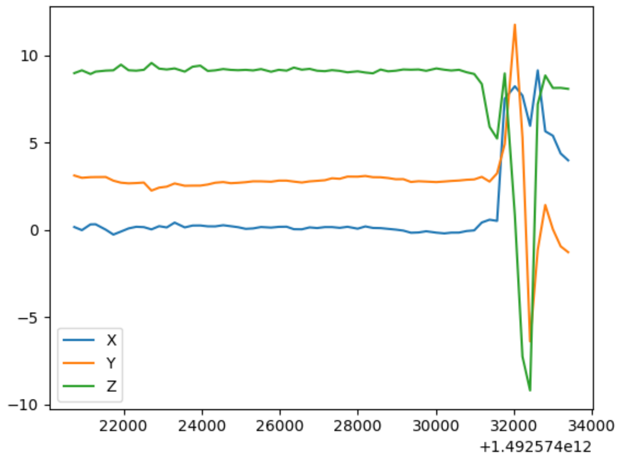


Fig. 4: A data graph generated from accelerometer data. After 10 seconds of standing, the fall creates a sudden change in the graph.

we examined the data graphs, we found that noises had been occurred at three different places: (1) at the beginning of the normal activities, different people had different types of start patterns, (2) at the end of the normal activity, when the participant heard a beep to fall, not all participants fell on the 20-second (in case of walk and jog) or 10-second (in case of stand, sit, and lie) mark, and (3) after fall, different situations arose for different participants as they pressed the 'STOP' button at different times. After analyzing the data graphs for all three axes, we created a filter that removes 1 second at the beginning of the activity, removes 0.5 seconds after a beep (when the fall is supposed to occur), and removes everything 1.5 seconds after the fall occurs.

We extracted 15 features from each window (sample), for each axis data. Figure 5 shows the list of features we extracted. Among these 15 features, we defined the first 11 features and selected the rest 4 features from existing literature. Figure 6 gives a pictorial view of our defined features.

Number of Up Peaks
Number of Down Peaks
Number of Cycles
Mean of Cycle Height
Mean of Cycle Width
Mean of Distance between Up Peak and Down Peak of a Cycle (Cycle Distance)
Standard Deviation of Cycle Height
Standard Deviation of Cycle Width
Standard Deviation of Cycle Distance
Custom Height
Custom Width
Mean of Axis Values
Standard Deviation of Axis Values
Root Mean Square (RMS) of Axis Values
MinMax of Axis Values

Fig. 5: List of Features.

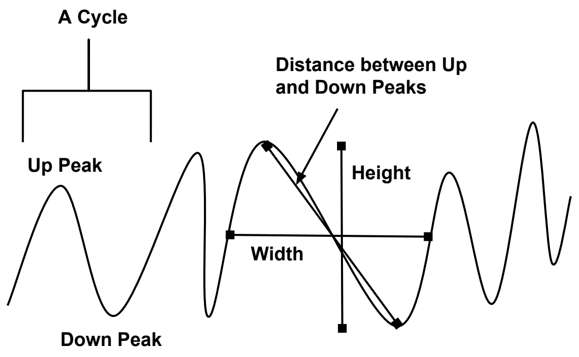


Fig. 6: Defining features in a data sample.

An up peak represents a highest point and a down peak represents a lowest point in a window. Generally, there are multiple up and down peaks in a window. Total number of up peaks and total number of down peaks in a window are the first two features in our feature set. A cycle is formed by one up peak and one down peak. We use number of cycles in a window as a feature. Figure 6 shows the height and width of a cycle. The mean of cycle height is calculated by adding the heights of all cycles and then dividing by the number of cycles. Similarly, the mean of cycle width is calculated by adding the widths of all cycles and then dividing by the number of cycles. The cycle distance refers to the Euclidean distance between the up and down peaks of a cycle. The mean of cycle distance is calculated by adding the distances of all cycles and then dividing by the number of cycles. We also use the standard deviations of cycle height, cycle width, and cycle distance, which quantify the amount of variation or dispersion of cycle heights, widths, and distances, respectively. The custom height feature retrieves the difference between the maximum height in a window and the average of other heights (which excludes the maximum of all heights in the window). Similarly, the custom width feature retrieves the difference between the minimum width in a window and the average of other widths (which excludes the minimum of all widths in the window). There are three features: mean, standard deviation, and root mean square (RMS) of axis values, where we do not use the concept of cycle and directly use the sensor values of an axis. Our last feature, the MinMax of axis values, is calculated by subtracting the minimum value from the maximum value of an axis.

VI. Experiments and Results

We evaluated our two-step fall detection method using data collected from five sensors: accelerometer, gyroscope, magnetometer, linear acceleration, and gravity. We experimented with five machine learning classifiers, namely, Support Vector Machine (SVM), K-Nearest Neighbors (KNN), Naive Bayes (NB), Decision Tree (DT), and Random Forest (RF). To train a classifier, we combined features from all three axes: X, Y, and Z of a sensor. Since we extracted 15 features from one axis of a sensor, a training sample consisted of 45 (3*15) features. Similarly, our test sample for a sensor also consisted of 45 features. For each sensor, we used 60% of our data for training and the rest 40% data for testing.

Our proposed fall detection method first performs multi-class classification, then it converts the multi-class predictions to binary decisions (fall or non-fall). We performed a set of preliminary experiments to determine the appropriate classes for multi-class classification. First we performed a 10-class classification, where classes were 'walk', 'jog', 'stand', 'sit', 'lie', 'fall from walk', 'fall from jog', 'fall from stand', 'fall from sit', and 'fall from lie'. After analyzing the data graphs, we found that data from 'stand', 'sit', and 'lie' activities were similar. Also, data from 'fall from stand', 'fall from sit', and 'fall from lie' were similar. Then, we performed a 6-class classification, where classes were 'walk', 'jog', 'stand-sit-lie', 'fall from walk', 'fall from jog', and 'fall from stand-sit-lie'. We obtained a slightly better result with 6-class classification. Hence, we performed 6-class classification at the first step in all our experiments. In the second step, a 'walk', 'jog', or 'stand-sit-lie' decision was converted to a non-fall decision. Similarly, a 'fall from walk', 'fall from jog', or 'fall from stand-sit-lie' decision was converted to a fall decision.

Table I shows the accuracy values of the five classifiers when our fall detection method is applied on data collected

TABLE I: Accuracy values obtained by our two-step fall detection method. Experiments are performed with five machine learning classifiers on data collected from five smartphone sensors.

Classifiers	Accelerometer	Gyroscope	Magnetometer	Linear Acceleration	Gravity
SVM	93.28%	95.65%	93.35%	92.33%	93.49%
KNN	91.69%	92.85%	91.02%	93.00%	90.71%
Naive Bayes	82.60%	84.47%	86.37%	84.33%	84.21%
Decision Tree	79.44%	92.23%	61.79%	89.66%	78.63%
Random Forest	88.93%	94.09%	89.03%	93.33%	91.95%

from five sensors. The Support Vector Machine (SVM) consistently achieves the highest accuracy values across all sensors. With the gyroscope sensor, Support Vector Machine obtains the maximum accuracy of 95.65%. The accuracy values of the K-Nearest Neighbor (KNN) are also consistent and over 90% across all sensors. The Random Forest (RF) and Decision Tree (DT) achieve accuracy values of 94.09% and 92.23%, respectively, with the gyroscope, however, their accuracy values are inconsistent across other sensors. The Naive Bayes (NB) consistently produces low accuracy values across all sensors.

Comparing the five sensors, the gyroscope produces the best accuracy of 95.65% when it is experimented with the Support Vector Machine. Among others, the accelerometer obtains its maximum accuracy of 93.28%, the magnetometer obtains its maximum accuracy of 93.35%, and the gravity obtains its maximum accuracy of 93.49%, all are experimented with the Support Vector Machine classifier. The linear acceleration obtains its maximum accuracy of 93.99% with the Random Forest classifier.

One issue we identified in our data is class imbalance problem, which is very common in machine learning research. The total number of instances in our fall class is far less than the total number of instances in non-fall class. This happened because we extracted multiple non-fall instances (windows) from a raw activity sample, however, we extracted only one fall instance from that raw activity sample. When class imbalance problem exists in the data, accuracy values do not give the complete picture of the the performance. To obtain a complete picture of the performance of our fall detection method, we generate ROC curves and use areas under ROC curves (AUCs) as evaluation metrics.

An ROC curve is defined by false positive rate (FPR) and true positive rate (TPR) as X and Y axes, respectively, which depicts relative trade-off between the true positives (benefits) and false positives (costs). True positive rate (TPR) is equivalent to the sensitivity and false positive rate (FPR) is equal to 1-specificity. Hence, the ROC curve is sometimes called the sensitivity vs. (1-specificity) plot. The area under the ROC curve (AUC) says how well the classifier can discriminate the two classes. The maximum possible value of AUC is 1, which means that the classifier perfectly distinguishes two classes. An AUC equal to 0.5 indicates that the classifier's performance is equal to a random guess.

Figure 7 shows the ROC curves obtained by several machine learning classifiers, for data collected from five sensors. We achieve the best AUC of 0.93 when Support Vector Machine (SVM) is applied on the gyroscope and gravity sensors. The SVM consistently demonstrates high performance with an AUC equal to 0.9 for the accelerometer and magnetometer sensors and an AUC equal to 0.89 for the linear acceleration sensor.

VII. Conclusion

In this paper, we develop a two-step fall detection method for smartphone users. Experiments performed on data collected from 10 smartphone users demonstrate that our proposed method is highly effective for fall detection. We achieve a maximum accuracy of 95.65% and a maximum area under ROC curve (AUC) of 0.93, with the gyroscope sensor and Support Vector Machine (SVM) classifier. In our experiments, the SVM consistently gives the best performance among five classifiers we tested. At the same time, the gyroscope sensor consistently demonstrates excellent performance. We believe that our findings will have a great impact assisting life in healthcare, and will raise the standards for current fall detection systems. Recently, researchers have started exploring deep learning techniques for human activity recognition. However, deep learning is still unexplored in the area of smartphone-based fall detection. Exploring different kinds of deep learning techniques for smartphone-based fall detection will be a great future research direction.

References

[1] M. C. Hornbrook, V. J. Stevens, D. J. Wingfield, J. F. Hollis, M. R. Greenlick, and M. G. Ory, "Preventing falls among community-dwelling older persons: Results from a randomized trial1," *The Gerontologist*, vol. 34, no. 1, pp. 16–23, 1994.

[2] "Injury prevention and control," Aug 2017. [Online]. Available: https://www.cdc.gov/injury/wisqars/.

[3] B. H. Alexander, F. P. Rivara, and M. E. Wolf, "The cost and frequency of hospitalization for fall-related injuries in older adults." *American journal of public health.*, Jul 1992. [Online]. Available: https://www.ncbi.nlm.nih.gov/pubmed/1609903

[4] "Ericsson mobility report," November 2018. [Online]. Available: https://www.ericsson.com/assets/local/mobility-report

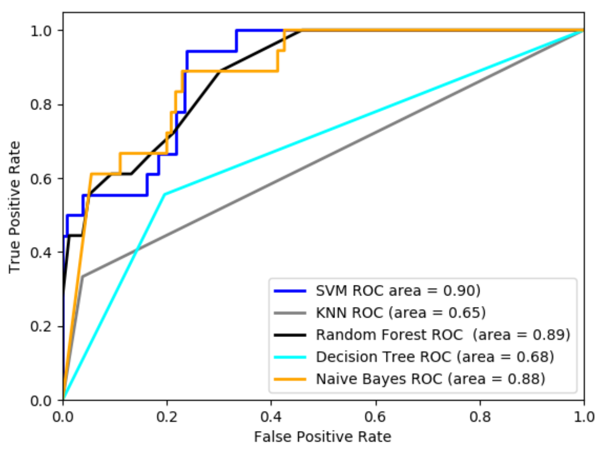
[5] "Location and sensors APIs," Jul 2017. [Online]. Available: https://developer.android.com/guide/topics/sensors/index.html

[6] X. Su, H. Tong, and P. Ji, "Activity recognition with smartphone sensors," *Tsinghua Science and Technology*, vol. 19, no. 3, pp. 235–249, June 2014.

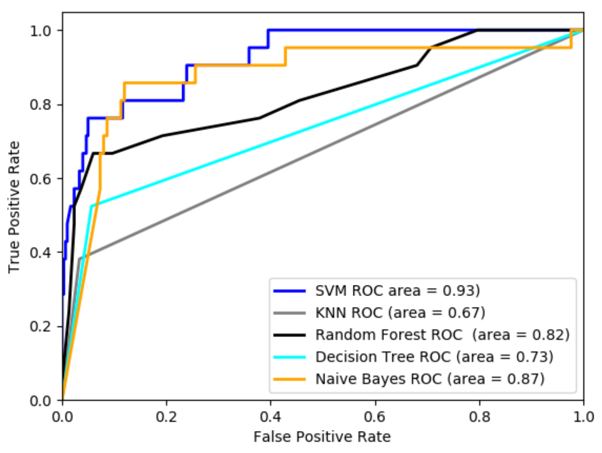
[7] J. Dai, X. Bai, Z. Yang, Z. Shen, and D. Xuan, "Perfalld: A pervasive fall detection system using mobile phones," in *2010 8th IEEE International Conference on Pervasive Computing and Communications Workshops (PERCOM Workshops)*, March 2010, pp. 292–297.

[8] Y. Lee, H. Yeh, K.-H. Kim, and O. Choi, "A real-time fall detection system based on the acceleration sensor of smartphone," *International Journal of Engineering Business Management*, vol. 10, 2018.

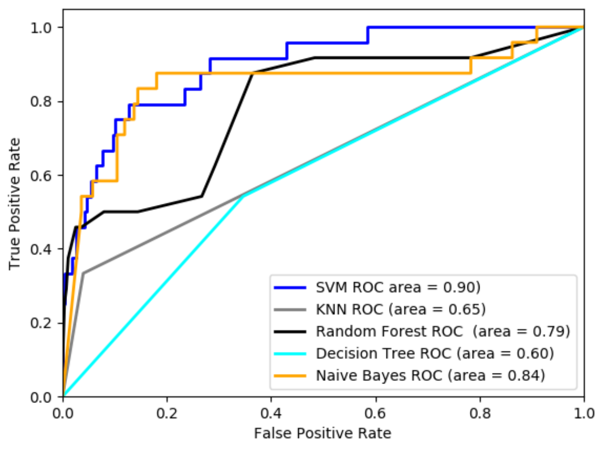
[9] Y. Hsu, K. Chen, J. Yang, and F. Jaw, "Smartphone-based fall detection algorithm using feature extraction," in *2016 9th International Congress on Image and Signal Processing, BioMedical Engineering and Informatics (CISP-BMEI)*, Oct 2016, pp. 1535–1540.



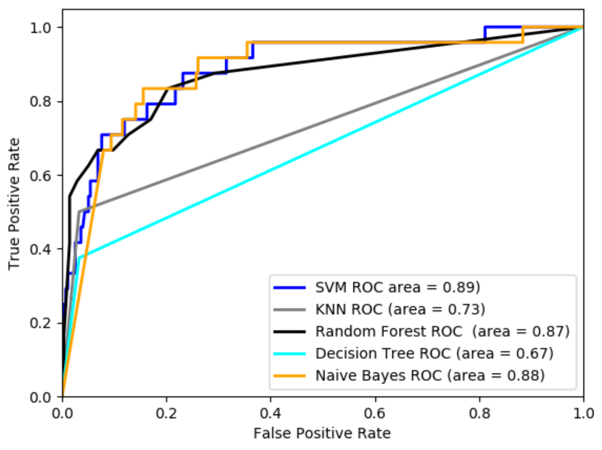
(a) Accelerometer



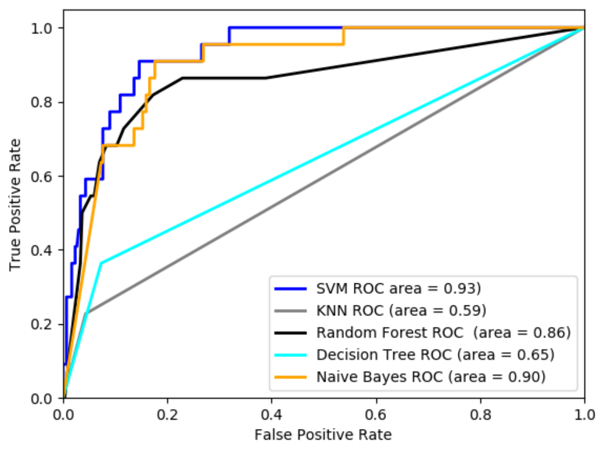
(b) Gyroscope



(c) Magnetometer



(d) Linear Acceleration



(e) Gravity

Fig. 7: ROC curves obtained by several machine learning classifiers, for data collected from five sensors.

[10] Q. Huynh, U. Nguyen, L. B. Irazabal, N. Ghassemian, and B. Tran, “Optimization of an accelerometer and gyroscope-based fall detection algorithm,” *Journal of Sensors*, vol. 2015, pp. 1–8, 04 2015.

[11] G. Chetty, M. White, and F. Akther, “Smart phone based data mining for human activity recognition,” *Procedia Computer Science*, vol. 46, pp. 1181 – 1187, 2015.

[12] D. Anguita, A. Ghio, L. Oneto, X. Parra, and J. L. Reyes-Ortiz, “Human activity recognition on smartphones using a multiclass hardware-friendly support vector machine,” in *Ambient Assisted Living and Home Care*, J. Bravo, R. Hervás, and M. Rodríguez, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, pp. 216–223.

[13] A. Bayat, M. Pomplun, and D. A. Tran, “A study on human activity recognition using accelerometer data from smartphones,” *Procedia Computer Science*, vol. 34, pp. 450 – 457, 2014.

[14] W.-Y. Deng, Q.-H. Zheng, and Z.-M. Wang, “Cross-person activity recognition using reduced kernel extreme learning machine,” *Neural Networks*, vol. 53, pp. 1 – 7, 2014.

[15] J. Wang, Y. Chen, S. Hao, X. Peng, and L. Hu, “Deep learning for sensor-based activity recognition: A survey,” *Pattern Recognition Letters*, 2018.

[16] C. A. Ronao and S.-B. Cho, “Human activity recognition with smartphone sensors using deep learning neural networks,” *Expert Systems with Applications*, vol. 59, pp. 235 – 244, 2016.

[17] S.-M. Lee, S. M. Yoon, and H. Cho, “Human activity recognition from accelerometer data using convolutional neural network,” in *2017 IEEE International Conference on Big Data and Smart Computing (BigComp)*, Feb 2017, pp. 131–134.

[18] M. Inoue, S. Inoue, and T. Nishida, “Deep recurrent neural network for mobile human activity recognition with high throughput,” *Artif. Life Robot.*, vol. 23, no. 2, pp. 173–185, Jun. 2018.

[19] Q. Li, J. A. Stankovic, M. A. Hanson, A. T. Barth, J. Lach, and G. Zhou, “Accurate, fast fall detection using gyroscopes and accelerometer-derived posture information,” in *2009 Sixth International Workshop on Wearable and Implantable Body Sensor Networks*, June 2009, pp. 138–143.

[20] U. Lindemann, A. Hock, M. Stuber, W. Keck, and C. Becker, “Evaluation of a fall detector based on accelerometers: A pilot study,” *Medical and Biological Engineering and Computing*, vol. 43, no. 5, pp. 548–551, Oct 2005.

[21] J. Chen, K. Kwong, D. Chang, J. Luk, and R. Bajcsy, “Wearable sensors for reliable fall detection,” in *Proceedings of the 27th Annual International Conference of the IEEE EMBS*, 2005, pp. 3551–3554.

[22] M. Prado, J. Reina-Tosina, and L. Roa, “Distributed intelligent architecture for falling detection and physical activity analysis in the elderly,” in *Proceedings of the Second Joint 24th Annual Conference and the Annual Fall Meeting of the Biomedical Engineering Society] [Engineering in Medicine and Biology*, vol. 3, Oct 2002, pp. 1910–1911 vol.3.

[23] A. Bourke and G. Lyons, “A threshold-based fall-detection algorithm using a bi-axial gyroscope sensor,” *Medical Engineering Physics*, vol. 30, no. 1, pp. 84 – 90, 2008.

[24] D. Lim, C. Park, N. Ho Kim, S.-H. Kim, and Y. S. Yu, “Fall-detection algorithm using 3-axis acceleration: Combination with simple threshold and hidden markov model,” *Journal of Applied Mathematics*, vol. 2014, September 2014.

[25] A. Yazar, F. Erden, and A. Cetin, “Multi-sensor ambient assisted living system for fall detection,” in *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, May 2014.

[26] C. Nadee and K. Chamnongthai, “Multi sensor system for automatic fall detection,” in *2015 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA)*, Dec 2015, pp. 930–933.

[27] W. Qu, F. Lin, and W. Xu, “A real-time low-complexity fall detection system on the smartphone,” in *2016 IEEE First International Conference on Connected Health: Applications, Systems and Engineering Technologies (CHASE)*, June 2016, pp. 354–356.

[28] T. Vilarinho, B. Farshchian, D. G. Bajer, O. H. Dahl, I. Egge, S. S. Hegdal, A. Lnes, J. N. Slettevold, and S. M. Weggersen, “A combined smartphone and smartwatch fall detection system,” in *2015 IEEE International Conference on Computer and Information Technology; Ubiquitous Computing and Communications; Dependable, Autonomic and Secure Computing; Pervasive Intelligence and Computing*, Oct 2015, pp. 1443–1448.

[29] E. Casilari and M. A. Oviedo-Jimnez, “Automatic fall detection system based on the combined use of a smartphone and a smartwatch,” *PLOS ONE*, vol. 10, no. 11, pp. 1–11, 11 2015. [Online]. Available: https://doi.org/10.1371/journal.pone.0140929

[30] T. Zhen, L. Mao, J. Wang, and Q. Gao, “Wearable preimpact fall detector using svm,” in *2016 10th International Conference on Sensing Technology (ICST)*, Nov 2016, pp. 1–6.