A Fall Detection Technique using Features from Multiple Sliding Windows

Sudarshan Pant[*], Jinsoo Kim[**], Sangdon Lee[***]

Abstract

In recent years, falls among elderly people have gained serious attention as a major cause of injuries. Falls often lead to fatal consequences due to lack of prompt response and rescue. Therefore, a more accurate fall detection system and an effective feature extraction technique are required to prevent and reduce the risk of such incidents. In this paper, we proposed an efficient feature extraction technique based on multiple sliding windows and validated it through a series of experiments using supervised learning algorithms. The experiments were conducted using the public datasets obtained from tri-axial accelerometers. The results depicted that extraction of the feature from adjacent sliding windows led to high accuracy in supervised machine learning-based fall detection. Also, the experiments conducted in this study suggested that the best accuracy can be achieved by keeping the window size as small as 2 seconds. With the kNN classifier and dataset from wearable sensors, the experiments achieved accuracy rates of 94%.

Keywords: fall detection | wearable sensor | multiple sliding windows | supervised learning

I. INTRODUCTION

Falls are one of the major health concerns as they not only impose the risk of physical injuries but also may lead to social, psychological and economic consequences. According to World Health Organization(WHO) [1], falls are the second leading cause of accidental or unintentional injury death throughout the world and 37.3 million falls are severe enough to require medical attention each year. Although falls affect people of all ages, the elderly people are highly vulnerable to fall-related consequences. A high precision fall detection system is therefore essential to reduce the likelihood of an elderly person being left unattended when a fall occurs. The main objective of the fall detection system is to identify the fall in real-time and notify the responders. Identifying the falls involve distinguishing falls from the activities of daily living(ADLs) based on the features of movement such as a change in acceleration and rotation of the body during various activities.

Fall detection techniques are broadly classified into wearable sensor-based and ambient sensor-based methods based on the data acquisition method and the type of data involved. Ambient sensor-based methods are commonly used in indoor environments while the wearable sensor-based methods are more suited for outdoor as well as personal use because of their portability [2]. Based on the detection approach the fall detection systems are classified into threshold-based and learning-based algorithms. Threshold-based approaches simply detect the falls by comparing the measured acceleration with a pre-defined threshold value of acceleration. While in machine learning-based methods, a model is trained with the labeled data and classifies the

* This work (Grants No. C0565317) was supported by Business for Cooperative R&D between Industry, Academy, and Research Institute funded by Korea Small and Medium Business Administration in 2017

* Member, Dept. of Multimedia Engineering, Graduate School, Mokpo National University
** Member, Graduate School of Computer Information Security, Chonbuk National University
*** Member, Dept. of Multimedia Engineering, College of Engineering, Mokpo National University

Manuscript : 2018. 08. 30
Revised : 2018. 11. 16
Confirmation of Publication:2018. 11. 17.
Corresponding Author : Sangdon Lee, e-mail : sdlee@mokpo.ac.kr

measured data based on the features. The sliding window method is commonly used for feature extraction in learning-based methods [3, 4]. Sliding windows are generally of two types; the fixed-sized non-overlapping sliding window(FNSW) and the fixed-sized overlapping sliding window(FOSW) [5, 6]. In existing sliding window based approaches [3, 7], features are extracted from single sliding window. This might result in missing important information about pre-impact or post-impact stage when it is beyond the current window being considered, which will be described in the Section III in detail. On the contrary, this paper proposes a fall detection method based on multiple sliding windows so that the information from the adjacent windows can also be considered. The key contribution of this paper is summarized as follows;

1) A peak based annotation of the sliding windows: A simple and effective annotation is important for fall detection as the fall sequences recorded during data acquisition are of different lengths.

2) Selection of significant features from multiple sliding windows using the information about pre-impact and post-impact stages of a fall:

The paper is organized as follows. Firstly, section II discusses some related works in the field of fall detection. Then section III explains the proposed approach for feature selection and fall detection. Section IV shows the experiments and the results. Finally, section V concludes the paper.

II. RELATED WORK

In the past, different wearable devices based on micro-electro-mechanical-systems(MEMS) sensors have been developed. Such devices use accelerometer and gyroscope data to distinguish between falls and ADLs. The overall fall detection system consists of data acquisition, fall detection, and alarm systems.

The fall detection systems implemented in the past can broadly be divided into threshold-based systems and learning-based systems. The threshold based approaches are based on the fact that the acceleration in falls is sharper than those in the ADLs. Lindeman et al. [7] proposed a threshold based fall detection system using accelerometer sensors embedded in a hearing-aid device. Li et al. [8] proposed a three-phase model where the fall event is identified in three phases; monitoring whether the user is static or dynamic, recognizing lying state, and judging whether the transition was intentional or not. Although threshold-based approaches have been able to detect falls [8, 9], manually defined thresholds do not generalize well for unseen subject hence result in a higher number of false alarms as noted by Bagala et al. [10]. On the other hand, in machine learning-based methods, the labeled data is used to train the models which classify the activities into falls or ADLs. Ojetola et al [11] used an accelerometer and a gyroscope and performed classification using C4.5 decision trees. Tong et. al. [12] used Hidden Markov Models and tri-axial accelerometer to detect and predict falls. Similarly, Dinh and Struck [6] developed learning-based algorithm for fall detection using the neural network and fuzzy logic to detect falls. Habib et al [13] demonstrated that a machine learning-based approach using SVM consumes the battery in a few hours hence implying that machine learning-based approaches are not suitable for use on the wearable devices.

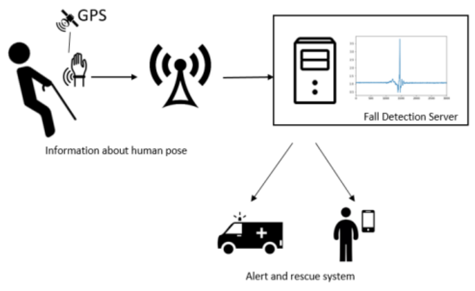


Fig. 1. Fall detection system overview

The sliding window based method is used for feature extraction in machine learning-based systems for fall detection. Selection of appropriate organization of a sliding window is crucial for finding temporal features from the sensor data. Due to lack of well-annotated standard dataset for comparison of the detection models, in the existing literature, various annotation techniques have been used for feature extraction from the sensor dataset. For instance, Putra et al. [7] studied the impact of fixed sized overlapping and non-overlapping sliding windows that included impact and post-impact phases of the falls. State machine based approach called Event-ML [4] used a threshold before the feature extraction.

In existing methods [3, 7], the features are extracted from the temporal windows with pre−impact, impact and post−impact stages of falls and the windows are annotated based on the overlapping with actual fall segments. However, such an approach is likely to result in more false negatives as some sliding windows may only contain the pre−impact or post−impact phase which is similar to some daily activities. Besides, the existing sliding window based approaches [3, 7] are restrictive in terms of feature selection as the features extracted from the sliding window are likely to be missing the significant information related to pre−impact phase and post−impact phase.

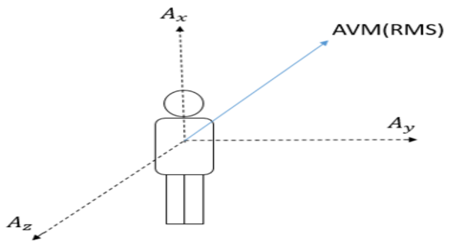
In the next section, we present the peak−based method for annotation of sliding windows which considers the neighboring windows for feature selection.

III. SLIDING WINDOWS BASED FALL DETECTION

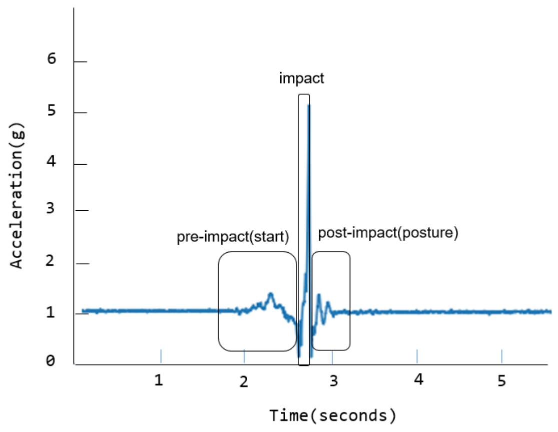
In the sliding window based method, the data sequence is first segmented into sliding windows of certain duration for feature selection and annotated for training a supervised learning model.

Acceleration signal is important information commonly used for analysis of human activity [14, 15]. Change is acceleration has been widely studied and considered as a significant characteristic of a fall [16]. Fig. 2 shows typical acceleration vector magnitude in falls. The acceleration vector magnitude AVM_t at certain a time t can be calculated using the Equation (1), where Ax_t, Ay_t and Az_t are the accelerations along the x−axis, the y−axis, and the z−axis as seen on Fig. 2 (a)

$$AVM_t = \sqrt{(Ax_t^{\,2} + Ay_t^{\,2} + Az_t^{\,2})} \qquad (1)$$



a) Acceleration vector magnitude(AVM)

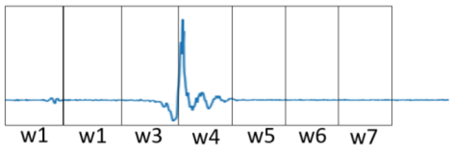


b) AVM at different stages of a typical fall sequence

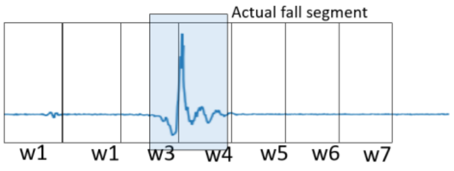
Fig. 2. Acceleration vector magnitude in fall

Based on the change in the resultant acceleration, a typical fall is divided into three stages; start, impact and posture [16]. During the start stage, the human body loses control and experiences a free−fall causing the acceleration to drop to a very small value which is close to 0g. When the impact with the ground occurs, there is a sharp change in acceleration and the acceleration stabilizes with little variation during the posture stage. The pre−impact, impact and post−impact stages of a typical fall can be visualized as in Fig. 2 (b).

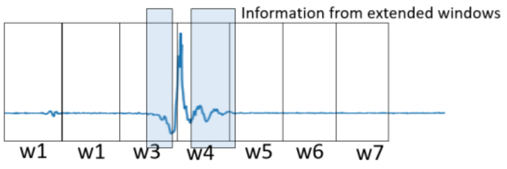
In the fixed−size non−overlapping sliding window based method windows sizes do not always align with the actual fall segments because of the varying duration of pre−impact, impact and post−impact phases of falls. When the features are selected from the sliding window, they represent the fall segment precisely only if the fall segment lies within the sliding window. However, in most cases, a fall segment is shared by multiple windows, and none of those represents the fall segment entirely. Fig. 3 shows sliding window alignment on a typical fall segment. In Fig. 3 (a), pre−impact part of a fall segment lies in the window ***w3*** while remaining part in window ***w4***. Missing pre−impact stage, which represents the onset of fall and where the acceleration decreased almost to zero due to free fall, the window ***w4*** is likely to be misclassified as an ADL despite the fact that it contains major portion of the fall. Fig. 3 (b) shows the actual fall segment which lies across multiple sliding windows.



a) Sliding window alignment on a typical fall segment



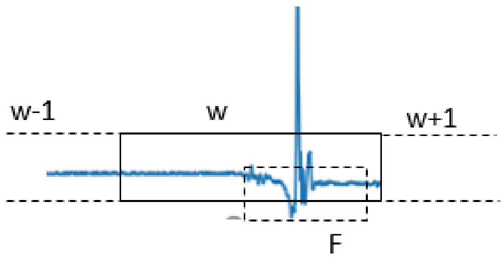
b) Actual fall segment not aligned with the sliding window.



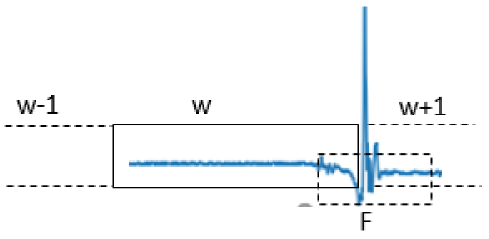
c) Feature selection using multiple windows
Fig. 3. Effect of window alignment in fall detection

In order to resolve this issue, we propose the extended window based feature selection technique where, in addition to the current sliding window, adjacent windows related to pre−impact and post−impact are used for feature selection as shown in Fig. 3 (c).

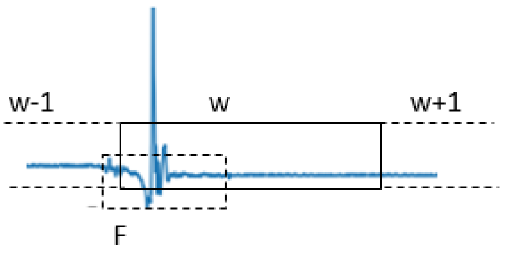
The duration of the fall event is not always the same. So, the length of the fall sequence is an important factor which affects the annotation of the windows. Although the fall sequence can be divided into pre−impact, impact, and post−impact, there is no clear distinction between them because the duration of each stage may vary. Since the annotation done during the data acquisition varies in length, the proper annotation is necessary before training the classifier. Thus the annotation has a high impact on the performance of fall detection. Particularly, the sliding window based fall detection is highly affected by the annotation method used to label the sliding windows. If the window does not coincide with the fall segment, the annotation becomes challenging. Following two methods of annotation can be applied in such cases.



a) A fall segment lies within the window



b) The smaller portion of fall lies within the window



c) The larger portion of the window lies within the window

Fig. 4. Different cases of window alignment with fall segment

In the sliding window based method, the windows do not coincide with the actual fall segment within the activity sequence. As shown in Fig. 4, the amount of overlap between the sliding window and the fall segment may vary leading to ambiguity in the annotation of the window. When a non-fall portion of the sliding window is larger than fall portion as in Fig. 4 (b), the features extracted from such a window will represent the ADL although it consists of a fall segment too. In another case, as shown in Fig. 4 (c) where the fall segment lies in two consecutive windows but the majority of the fall segment lies within the current window. In such cases, the windows can be annotated in different ways as follows.

1) Labeling the window as fall if any sample within the window is fall. This method is suitable when the data recorded from the sensors have a clear distinction between the pre-impact, impact and post-impact, which however is impossible because there is no clear criterion for marking the start or the end of the fall event. Hence the recorded fall segments always have some non-fall characteristics in the beginning and end of the sequence. Thus annotation of windows based on this method leads to the faulty training of the classifier.
2) Labeling based on the ratio of fall overlap; the window with a higher number of fall samples being labeled as fall. Another approach is to annotate the window as a fall if the major portion of the fall lies within the sliding window. This approach is likely to represent the window correctly because the larger number of samples from fall sequence contribute to the features. However, in most cases, actual fall is only a small segment in that fall sequence.
3) Labeling based on a specific feature of the fall. This is the proposed method in this paper where the window is annotated based on the presence of impact. Regardless of the duration of the fall segment, the impact phase is the most distinctive feature in every fall and being related to a single sample it does not vary in length.

Annotation of the sliding windows, extraction of features for training the model, and finally fitting and evaluation of the classification is done as discussed below.

A. Annotation of sliding windows

In the Cogent dataset [17], the data is gathered and a sequence of all the events are saved as comma separated values and each sample is annotated showing whether the sample belongs to fall, ADL or near-fall. For this study we classified the dataset into fall and non-fall only, thus near-fall segments are labeled as ADLs. Annotation of sliding windows is performed in two steps as follows.

1. *Peak Selection*

When a fall sequence is divided into sliding windows, the annotation of such windows is done based on the peak value of acceleration vector magnitude which represents the impact stage of a fall. The peaks in all fall segments for the given Cogent data sequence is extracted by finding the maximum value of acceleration vector magnitude in those fall segments. The peak selection algorithm for the Cogent dataset is as shown in Algorithm 1.

Algorithm 1 Peak selection for annotation of sliding windows

```
Input: a data frame with sensor data for a subject
Output: list of peaks in all activities.
Initialize a list P
while the sequence has next element A
    if A is labeled as fall
        find max value m.
    else
        add m to list P
return the list P
```

2. *Annotation*

Sliding windows of various sizes are created and annotated as a fall or an activity of daily living based on the position of the peak. Although the activity sequences Cogent dataset are labeled, the segments labeled as fall are larger than the actual fall event as shown in the Fig. 5. Thus, the sequences annotated as a fall also include the data corresponding to non-fall states before and after the fall event. So sliding windows need to be annotated in such a way that the fall events are represented more accurately. The windows were annotated as fall(F) and ADL(D) with following criteria:

1. If the segment is labeled as ADL all the sliding windows are annotated as ADL.
2. If the segment is labeled as a fall, then among the sliding windows that contain the peak were annotated as fall and rest of the windows were annotated as ADL.

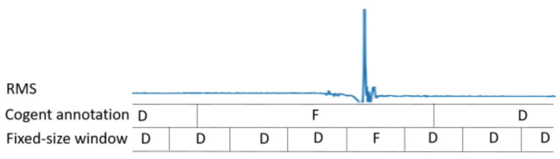


Fig. 5. Peak based annotation of sliding windows.

B. Feature Extraction

Feature selection in the sliding window based approach involves data-segmentation as the initial step. From each segment, features are extracted and different machine learning models were trained with those labeled data segments. Following features; A, B, C, D are used in the experiments as there is a sudden change in the accelerometer during fall events.

A. RMS (Root Mean Square) of the acceleration vector magnitude within the window.
B. Three minimum and maximum values before the peak
C. Three minimum and maximum values after the peak
D. Difference between the peak acceleration value in the window and the smallest value of acceleration before the peak.

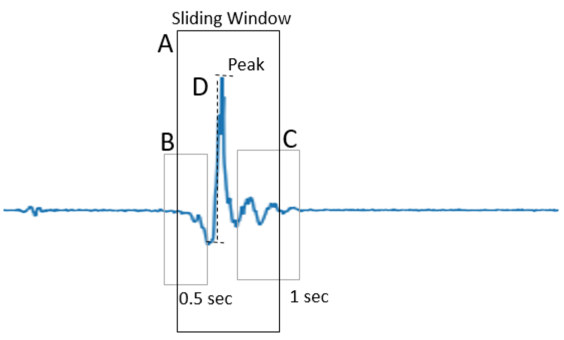


Fig. 6. Feature selection from significant sections of pre-impact, impact and post-impact stages of a fall.

In Fig. 6, feature A represents the current window in terms of RMS of the acceleration vector magnitude of all the samples in it. Since the temporal window often excludes the significant information about the fall when it partially overlaps with the fall sequence. In order to include the significant changes in the acceleration signal pertaining to the pre-impact and post-impact phases, the features A, B, C and D were extracted. We conducted a series of experiments with various combination of the features, and concluded that using the feature B has a little impact on the performance of fall detection. Hence we decided to use A, C, and D as the features for fall detection.

C. Threshold-based filtering

We applied threshold in order to narrow down the scope of classification by sending only those activities which have acceleration higher than a certain threshold. Since the falls are characterized by high acceleration value, the threshold can be used to detect falls but some activities of daily living like jumping may often produce high acceleration signals resulting in false positives during classification.

In case of real-time online fall detection systems, applying threshold not only reduces the data to be held for feature extraction, it also reduces the bandwidth usage by sending the data to the server only when there is some activity with higher acceleration. The threshold of 1.8 was taken for this study as it was found to be the optimal threshold for the active state [4].

D. Fall detection using supervised learning

The supervised learning-based fall detection involves the classification of the sensor data based on the labeled data and is conducted in two phases; the training phase and a classification phase. The training phase involves fitting the models with the labeled data and in the classification phase, the unseen data is introduced to the trained model for classification.

Ⅳ. EXPERIMENTS

The data was preprocessed for annotation of the windows and for extraction of required features from raw sensor data available in the public dataset.

For classification, kNN implementation from Scikit-learn library [18] was used as machine learning models for binary classification between falls and ADLs.

After the sliding windows are classified using the trained models, the performance evaluation is done using cross-validation. We adopted the leave-one-subject-out cross-validation method where the sensor data obtained from one subject is left out during training phase so that it can be used as test data for evaluation and the process is iterated for all subjects. Validation is done against the true labels obtained from the annotation of the windows. When the window labeled as fall is classified as fall by the classifier, it is considered true positive, and when the ADL window is classified as an ADL, it said to be a true negative. Thus there are four cases of the outcome namely True Positive (TP), False Positive (FP), True Negative (TN) and False Negative (FN). The performance of the classifier is measured in terms of precision and recall values which in turn can be represented as F1-score which is a harmonic mean of precision and recall.

$$Precision = \frac{TP}{TP+FP} \qquad (2)$$

$$Recall = \frac{TP}{TP+FN} \qquad (3)$$

$$F1-Score = \frac{2 \times Precision \times Recall}{Precision + Recall} \qquad (4)$$

1. *Dataset*

The Cogent dataset consists of the readings from two wearable devices worn on the chest and thigh, however, the data used for this study was from the device worn on the chest because the chest is considered as the best position for sensor placement for fall detection [19].

Table 1. Characteristics of subjects in Cogent dataset.

Gender		Age	Height	Weight
Female	Male	(years)	(cm)	(kg)
9	37	23.5 ± 5.5	172.7± 7.7	67.7± 12.8

As shown in Table 1, the cogent dataset was collected from 46 subjects including 9 females and 37 males with age 23.5 ± 5.5 years. That dataset has 644 falls and 1,196 ADL events altogether. This dataset was chosen for the experiments mainly because it includes different types of activities and the annotation of each sample makes it easy to annotate the windows of different sizes.

2. *Parameter selection*

The parameter ***k*** for this experiment using the kNN classifier was chosen based on the preliminary experiments with different values of ***k***.

The performance of kNN classifier for the 2-second non-overlapping sliding window was observed for values different values of ***k***. Table 2 shows that 5 is the optimal value for classification. Although there is not much difference in the accuracy for 5, 7 and 9, the recall has the highest value for ***k*** = 5.

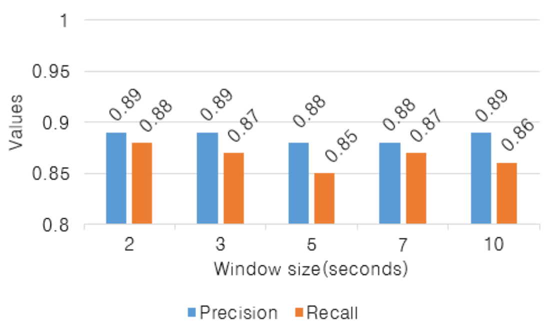
Table 2. Performance of kNN with different values of k on the Cogent Dataset

	K=3	K = 5	K=7	K = 9
Precision	92.67	93.39	93.21	93.39
Recall	94.42	94.75	94.54	94.54
F1-score(%)	93.03	93.60	93.29	93.39

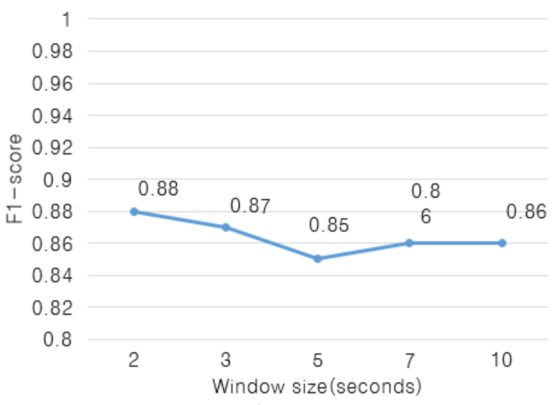
3. *Results and discussion*

In this experiment, we analyze the performance of different machine learning classifiers based on sliding windows of different sizes. The experiments were initially conducted without using the threshold filtering. The features were extracted from multiple sliding windows and classification performed by training different machine learning models as discussed in section III.

Fig. 7 (a) shows the precision and recall scores for different non-overlapping window sizes using kNN classifier. Precision is high for smaller windows but the recall is highest for the two-second window. F1-score is highest for the two-second window size as shown in Fig. 7 (b).



a) Precision and recall



b) F1-Score

Fig. 7. Performance comparison for FNSW using kNN classifier with different window sizes.

Without applying a threshold, some activities which are fall like but with a lesser impact on the ground are incorrectly classified as falls resulting in many false positive. In order to reduce false positives, we applied a threshold first so that only the acceleration signal for the active state is used for classification.

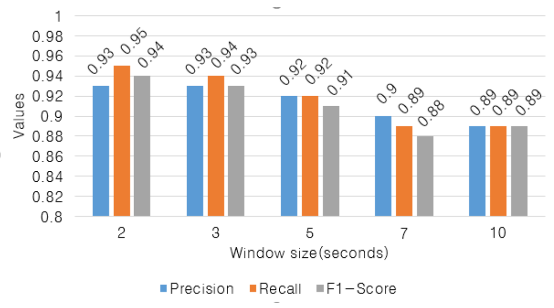


Fig. 8. Classification performance using kNN classifier with different window sizes using a threshold.

Fig. 8 shows the comparison of different performance measures for varying window sizes. By applying the threshold, the accuracy has improved compared to the Fig. 7. The tendency of the decreasing performance was the same with the non-threshold case as the window size is increasing.

Table 3 shows performance of different classifiers for various sliding window organizations. The recall (false negative rate) is an important performance measure for fall detection as the goal is to reduce the misclassification of a fall as an ADL while reducing the false positives. It can be seen in the table 3 b) that recall is also reduced by using threshold before feature extraction. The performance measures for the experiments using threshold(FNSW-TH) are compared to those using fixed-size non-overlapping sliding window (FNSW), fixed-size overlapping windows with 25%, 50%, 75% overlap denoted by FOSW-25, FOSW-50, FOSW-75 respectively. Table 3 a), b) and c) show the comparison of precision, recall and F1-score for different overlap percentages while using three different classifiers. It can be seen that precision is highest when non-overlapping window is used and there is not much impact of overlap. On the other hand using a threshold resulted in higher accuracy.

Table 3. Performance of classifiers for different sliding window organizations.

a) Precision

Method	kNN	Decision Trees	Naïve Bayes
FNSW-TH	93.39±10.0	92.69±7.84	80.9±17.57
FNSW	89.32±11.66	85.36±10.57	74.47±15.66
FOSW-25	89.06±9.36	84.60±11.47	74.99±16.36
FOSW-50	88.32±11.39	84.23±10.92	74.22±15.55
FOSW-75	86.60±10.60	85.18±10.28	74.17±15.46

b) Recall

Method	kNN	Decision Trees	Naïve Bayes
FNSW-TH	94.75±11.91	89.85±13.38	77.8±25.91
FNSW	87.94±15.33	86.60±0.14	85.04±22.24
FOSW-25	87.99±16.30	83.19±15.93	85.69±21.92
FOSW-50	87.83±15.31	82.92±17.13	85.37±21.87
FOSW-75	85.15±1664	83.31±14.18	85.37±21.98

c) F1-score

Method	kNN	Decision Trees	Naïve Bayes
FNSW-TH	93.60±9.79	90.52±8.57	77.56±21.11
FNSW	87.77±11.88	85.25±10.75	77.93±17.29
FOSW-25	87.49±11.83	82.76±11.28	78.51±17.40
FOSW-50	87.11±11.50	82.49±12.10	78.03±16.98
FOSW-75	84.76±12.19	83.43±9.98	78.00±17.07

Also, the results show that the use of non-overlapping windows produces better results as compared to the overlapping windows.

We compared performance of our approach with the existing study using the kNN classifier. For the same experimental condition with the existing approaches, we used the same dataset (the Cogent dataset) and the same validation method(LOOCV). Table 4 shows the results of performance(F1-score) comparison with the Putra's works[4,13] which are the most recent works.

Table 4. Performance Comparison with the exisiting study

	Our approach	Putra et. al [4, 13]
FNSW-TH	93.60±9.79	94 ± 8.8 (EvenT-ML)
FNSW	87.77±11.88	79.7 ±11.5
FOSW-25	87.49±11.83	76.1 ± 11.8
FOSW-50	87.11±11.50	72.5±10.7
FOSW-75	84.76±12.19	60.9 ± 9.5

For the FOSW, our approach outperforms Putra's approach by 19.5%(FOSW-25) ~ 39%(FOSW-75). For the FNSW, our approach recorded 10% performance gain. EvenT-ML [4] uses a threshold to initiate a classification of falls based on multiple peak detection only when acceleration exceeds a given threshold. Using the kNN classifier, our approach obtained a comparable performance to the Event-ML as shown in the Table 4. However, EvenT-ML approach requires more complex feature set. Furthermore, it needs features from the whole pre-impact stage too, hence requiring larger buffer to hold all the data obtained from that stage. On the contrary, our approach uses more simple feature set in the sense of the number of features, and the calculation overheads. Our approach requires to manage only the minimum acceleration prior to the threshold, both the peak acceleration and the RMS of resultant acceleration magnitudes from the current sliding window, and finally the minimum and maximum values after the peak across the window boundary. This helps to reduce the computation overhead needed to obtain the necessary features, and also the volume of the data transferred from the wearable device to the fall detection server.

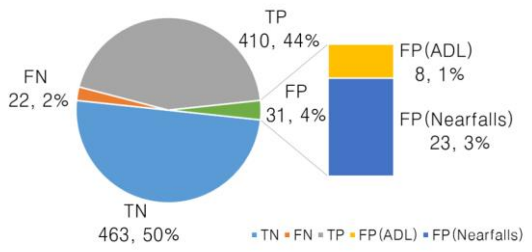


Fig. 9. Classification results showing the percentage of near-fall cases leading to false positives.

Detailed analysis to the fail cases to detect falls revealed that the near-fall activities were frequently misclassified to be falls. In the experiment with 2-second non overlapping sliding window using kNN classifier, 31 activities out of 926 activities (4%) were false positives. Among 31 false positives cases, 23 were found to be originally annotated as near falls in the Cogent dataset. Fig. 9 shows detailed analysis of near-fall cases leading to false positives. It reveals that 3% of the test activities were near-falls, which led to majority of false positive results during the binary classification. The classifier was able to classify 147 out of 170 total near-falls in the test dataset correctly. We found that detecting the near-fall cases correctly is a crucial factor to enhance the accuracy of fall detection.

V. CONCLUSION

This paper proposed the peak based annotation method and a novel feature extraction which considers use of multiple sliding windows. We have analyzed the impact of applying the multiple sliding windows for fall detection using different supervised machine learning algorithms. Extensive experiments were conducted with non-overlapping as well as overlapping windows of various sizes.

The proposed feature selection method and kNN Classifier(k=5) identified the falls against the ADLs with an average F-1 score of 87.77% without applying the threshold acceleration, and 93.60% when the threshold is applied before feature selection. These results clearly show that our proposed approach improves performance by selecting the representative significant features considering the adjacent sliding windows together.

This paper has mainly focused in exploiting features from multiple windows. Further study includes the use of different features that can further enhance the accuracy. We will also explore the effects of applying more advanced learning techniques including ANN and deep learning methods.

REFERENCES

[1] WHO. Falls (2018). http://www.who.int/en/news-room/fact-sheets/detail/falls (accessed Aug., 27, 2018).

[2] Mao A.; Ma X.; He Y.; Luo J.; " Highly portable, Sensor-Based System for Human Fall Monitoring, Sensors," *Sensors*, 2017.

[3] He, J.; Bai, S.; Wang, X.; " An unobtrusive fall detection and alerting system based on Kalman filter and Bayes network classifier," *Sensors*, 2017.

[4] Putra, P.; Brusey, J.; Gaura, E.; Vesilo, R.; " An Event Triggered Machine Learning Approach for Accelerometer-Based Fall Detection," *Sensors*, 2018.

[5] Diep, N.; Pham, C.; Phuong, T.; "A classifier based approach to real-time fall detection using low-cost wearable sensors.," *Proc. Of International Conference on Soft Computing and Pattern Recognition(SoCPaR)*, pp.105-110, 2013.

[6] Dinh, C.; Struck, M.; "A New Real-time Fall Detection Approach Using Fuzzy and neural network," *Proc. Of International Conference on Wearable Micro and Nano Technologies for Personalized Health*, pp.57-60, 2009.

[7] Putra, P.; Vesilo, R.; "Window-size impact on detection rate of wearable-sensor based fall detection using supervised machine learning.," *Proc. Of IEEE Life Sciences Conference*, pp.21-26, 2017.

[8] Li, Q.; Stankovic, J.; Hanson, M.; Barth, A.; Lach, J.; Zhou G.; "Accurate, Fast Fall Detection using Gyroscopes and Accelerometer Derived Posture Information," *Body Sensor Network*, International Workshop on Wearable and Implantable Body Sensor Networks, pp.138-143, 2009.

[9] Lindemann, U.; Hock, A.; Stuber, M.; Keck, W.; Becker, C.; "Evaluation of a Fall Detector Based on Accelerometers: A Pilot Study," *Med. Biol. Eng. Coput.*, vol.43, pp.548-551, 2005.

[10] Bagala, F.; Becker, C.; Cappello, A.; Chiari, L.; Aminian, K.; Hausdorff, J.; Zijlstra, W.; Klenk, J.; "Evaluation of Accelerometer-based Fall Detection Algorithms on Real-world Falls.," *PLoS ONE*, 2012.

[11] Ojetola, O.; Gaura, E.; Brusey, J.; "Fall Detection with Wearable Sensors-SAFE (SmArt Fall dEtection)," Proc. Of International Conference on Intelligent Environments(IE), pp.318-321, 2011.

[12] Tong L.; Song, Q.; Ge, Y.; Liu M.; " HMM-Based Human Fall Detection and Prediction Method Using Tri-Axial Accelerometer.," *IEEE Sens. J.*, vol.13, pp.1249-1256 vol.1, 2013.

[13] Habib, M.A.; Mohktar, M.S.; Kamaruzzaman, S.B.; Lim, K.S.; Pin, T.M.; Ibrahim, F.; " Smartphone-Based Solutions for Fall Detection and Prevention: Challenges and Open Issues," *Sensors*, vol.14, pp.7181-7208, 2014.

[14] Tae Woong Kim, "Group Behavior Pattern and Activity Analysis System Using Big Data Based Acceleration Signals," *Smart Media Journal*, vol.6, no.3, pp.83-88, 2017.

[15] Younghun Lee, Yongil Kim, "Design of Building Biomertic Big Data System using the Mi Band and MongoDB," *Smart Media Journal*, vol.5, no.4, pp.124-130, 2016.

[16] Kangas, M.; Vikman, I.; Wiklander, J.; Lindgren, P.; Nyberg, L.; Jamsa, T.; Sensitivity and Specificity of Fall Detection in People Aged 40 Years and Over.," *Gait Posture*, vol.29, no.4, pp. 571-574, 2009.

[17] Ojetola, O.; Gaura, E.; Brusey, J.; "Data Set for Fall Events and Daily Activities from Inertial Sensors.," *Proc. Of ACM Multimedia Systems Conference*, pp.243-248, 2015.

[18] Pedregosa, F.; Varaquaux, G.; Gramfort, A.; Michel, V.; Thirion, B.; Grisel, O.; Blondel, M.; Prettenhofer, P.; Weiss, R.; Douborg, V.; Vanderplas, J.; Passos, A.; Cournapeau, D.; Brucher, M.; Perrot, M.; Duchesnay, E.; "Scikit-learn: Machine Learning in Python.," *Journal of Machine Learning Research*, vol.12, pp. 2825-2830, 2011.

[19] Gjoreski, M.; Lustrek, M.; Gams, M.; "Accelerometer Placement for Posture Recognition and Fall Detection.," *Proc. Of International Conference on Intelligent Environment(IE)*, pp.47-54, 2011.

Authors

Sudarshan Pant

He received the B.S., and M.S., in Multimedia Engineering in Mokpo National University, Korea in 2010, and 2012, respectively. He worked as a Software Engineer from 2012 to 2016. He is currently a Ph. D student in the Department of Multimedia Engineering at Mokpo National University. His research interests include Computer Vision, Machine Learning, and Data Science.

Jinsoo Kim

He received the B.E. degree in computer engineering from Seoul National University, Korea in 1985, and the M.S. degree of Computer Science from Yonsei University, Seoul, Korea in 1996. He is studying the information protection for Ph.D. at Chonbuk National University. Main areas of interest are the personal information protection, big data analysis, distributed computing system.

Sangdon Lee

He received his B.S., M.S., and Ph.D. degrees in Computer Engineering from the Department of Computer Engineering, Seoul National University, Korea in 1984, 1986 and 1996 respectively. He worked at Research and Development Group in Korea Telecom, Seoul, Korea for 10 years before joining to Mokpo National University. He is currently a professor of the Department of Multimedia Engineering and the Department of Information Security. His research interests include big data management, multimedia services, intelligent data processing in IOT environments.