Novel Multi-Layer Ensemble Algorithm for Fall Detection using wrist-worn devices

Purab Nandi
Department of EEE
BITS Pilani, K.K Birla Goa Campus
Goa, India
p20200056@goa.bits-pilani.ac.in

K.R Anupama
Department of EEE
BITS Pilani, K.K Birla Goa Campus
Goa, India
anupkr@goa.bits-pilani.ac.in

Arav Jain
Department of EEE
BITS Pilani, K.K Birla Goa Campus
Goa, India
f20201452@goa.bits-pilani.ac.in

Siddhart Paliwal
Department of EEE
BITS Pilani, K.K Birla Goa Campus
Goa, India
f20201453@goa.bits-pilani.ac.in

OM Tayal
Department of EEE
BITS Pilani, K.K Birla Goa Campus
Goa, India
f20190707@goa.bits-pilani.ac.in

Himanish Agarwal
Department of EEE
BITS Pilani, K.K Birla Goa Campus
Goa, India
f20200308@goa.bits-pilani.ac.in

***Abstract*— Unattended falls of the elderly are a severe health hazard. Several efforts have been made to use Machine Learning and Deep Learning (ML/DL), Internet of Things (IoT), and wearables for Fall detection in the elderly. This paper presents a custom wrist-worn device built using Qualcomm Snapdragon 820c to detect falls. We have used Dew computing, where the data is processed on the wearable device. This system reduces the latency, and issues related to network dysconnectivity are solved. To ensure minimum latency and maximum accuracy, we have used our multi-layer ensemble of ML/DL algorithms. We call this ensemble algorithm, which we developed as variable weight ensemble algorithm – D(VWE(D)). This ensemble algorithm runs on the End device, equipped with a medical-grade Inertial measurement unit (IMU) and heart-rate sensors. The data is collected from the sensors. The statistical features are extracted from the sensors; the data is pruned using Shapley values. The deep learning algorithms are part of the ensemble, Convolution neural network (CNN), and Multi-layered perceptron (MLP), which are pruned using a modified version of the MIT lottery ticket hypothesis. We have obtained an accuracy of 98.2% with specificity and precision of 100%; also, an F1 Score of over 97% indicates that the results are outstanding. In this paper, we have analysed the accuracies and latency on 820c while using the ensemble algorithm and also analysed the accuracies and latencies of the individual components in the ensemble algorithms.**

Keywords*— *Fall Detection, SoCs, Ensemble Techniques, Pruning, Wearables

I. Introduction

Unintentional fall injuries are the fifth leading cause of death in the elderly (after cardiovascular, cancer, strokes, and pulmonary disorders). Two-thirds of the falls result in death [1]. Even though some of the falls may not be very hard falls, leading directly to death, the late response would result in the elderly trying to move themselves or may lose consciousness. This condition results in further complications, which increases the morbidity rate. Quick response to falls is the primary solution to this problem.

India is ageing and ageing at a very high rate, unlike other developed countries where the ageing population is usually healthy; in India, the geriatric population suffer from multiple health issues. Also, with the growth of nuclear families, the elderly are left to fend for themselves or relegated to over- crowded old-age homes, where little to no medical attention is available for emergencies such as falls. The solution is to use technology.

In recent years, research has grown exponentially in the field of fall detection and prediction using the Internet of Things (IoT) [3]and ML/DL algorithms [2]. The methodology used by IoT is usually to use readily available sensors on smartphones, collect the IMU data and send it to the cloud for processing. With the aged Geriatric population in India, this methodology has several issues, such as:

1) Several of them may reside in rural and semi-urban areas where the internet connectivity may not be well established, maybe sketchy at best.
2) Several of the elderly own basic cell phones rather than smartphones and are uncomfortable using smartphones.
3) Even if their connections are good, the latency in sending the data to the cloud and sending the prediction to the medical aide from the cloud is usually very high.

To solve these issues, we have developed a low-cost wrist- worn device that will not only collect data but also make a prediction and sound an alarm in case of falls. Here, the latency is negligible, as can be visualized in a later section on results and discussions.

The architecture of the system is shown in Figure 1.

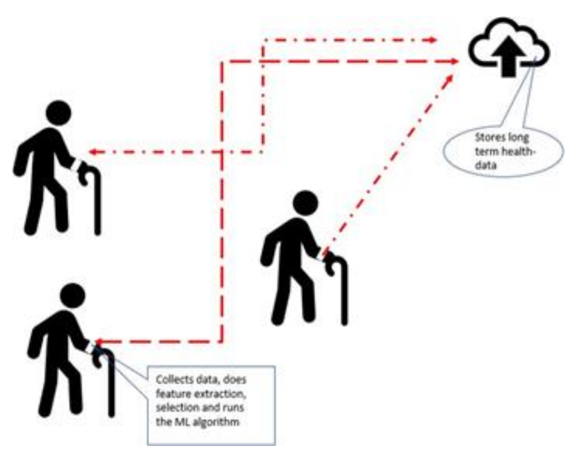


Fig. 1: Architectural model of wearable fall detection

Multiple ML algorithms and DL algorithms [4], [5], [6] ,[7],[8] and [9] have been individually tried for Fall prediction on the cloud with varying accuracies and latencies. This paper gives details of these algorithms. We initially developed a Stacking and two voting-based algorithms which could run on the custom device at latencies less than 1 ms with accuracies of about 97% [10]. We have modified these algorithms to improve the accuracy and sensitivity while retaining low latencies by incorporating DL algorithms into the ensemble. We have also tried multi-layered ensemble algorithms with successful improvement in accuracy with 100% specificity (No ADL is classified as a fall).

The sections are organised as follows: Section 2 describes the architecture of the wrist-worn device. Section 3 gives a brief on the data-gathering process. Section 4 describes the novel ensemble algorithm developed based on the multi-layer voting system. Section 5 presents the results and discussions, and finally, we conclude with section 6.

II. Architecture of the End-Device

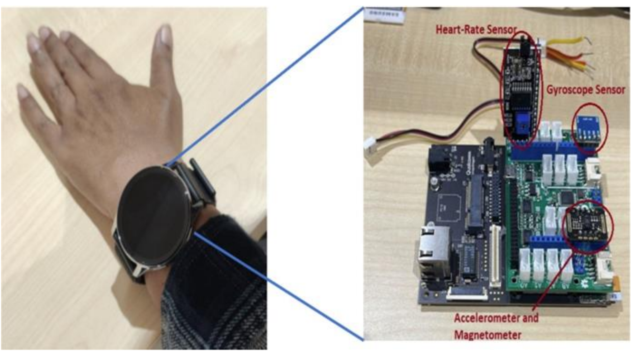


Fig. 2: Prototype diagram of the wearable device

Figure 2 shows the complete block diagram of the end device used. The core of the wearable system comprises a powerful SoC, which in our case was Snapdragon 820c [10]. The Soc is interfaced with an industrial standard Accelerometer and gyroscope sensors.

The core of the wearable system comprises a powerful SoC, which in our case was Snapdragon 820c . The Soc is interfaced with an industrial standard Accelerometer and gyroscope sensors. For our wearable device we used LSM6DSO16IS from STMicroelectronics [12] which has 3D accelerometer and 3D gyroscope with intelligent processing unit. For magnetometer we used LIS3MDL which is a ultra low-power magnetometer for wearable devices[13].

The IMU data is sampled at a rate of 20 Hz. The IMU data is augmented with the heart rate data as a person may experience a spike in the heart rate during the fall due to increased panic levels. Also, a skin temperature sensor is included as there is a possibility that the temperature may rise during the fall. The data collected on the end device is stored on the SD card and sent to the cloud for long-term health monitoring. We are running a basic vanilla Linux kernel on the System on Chip (SoC), and the features are extracted in real-time, and a decision is made based on the prediction of the ensemble algorithm. An alarm is sounded in case of a fall.

III. Data Collection and Methodology

Many publicly available datasets exist for "Fall Detection", are available such as MobiFall [15] , SisFall [16], SmartFall [17], SmartWatch [18], and Notch [19] but, there are various associated issues such as :

- Few datasets have multiple sensor readings, they primarily have only accelerometer-based readings.
- There is a lack of diversity in several datasets or information regarding demographic diversity of the volunteers is not available to train the model to work across different population demographics.

Hence, To address the above mentioned issues we collected we collected data from a diverse population of 41 people performing multiple activities (Activities of Daily Living (ADLs) and falls) for five trials per person. The cleaned and averaged value parameters were stored. The Dataset is available at https://zenodo.org/badge/latestdoi/517690954

TABLE I: List of Activities of Daily Living (ADLs) performed by volunteers

Activities of Daily Living (all activities had been performed with 5 trails each)

Stationary Movement	**Period**	**Standard Movement**	**Period**	**Sporting Movements**	**Period**
Slowly sitting on chair	30 sec	Walking slow	2 min	Walking quickly	2 min
Rapidly sitting on chair	30 sec	climbing up slowly	2 min	Jogging	2 min
Nearly sitting on chair and getting up	30 sec	climbing down slowly	2 min	Jumping	30 sec
Swinging hands	2 min	Lying on back and getting up slowly	30 sec	climbing up fast	2 min
Lying on Bed	2 min	Lying on back and getting up quickly	30 sec	climbing down fast	2 min
transition from sideways to one's back while lying	30 sec				

TABLE II: List of Fall Activities performed by volunteers

Hard and Soft Falls (all activities had been performed with 5 trails each)

Hard Falls	**period**	**Soft Falls**	**period**
Forward Fall landing on Knee	40 sec	Forward Fall	40 sec
Seated on Bed and falling on ground	40 sec	Right Fall	40 sec
Forward Fall body weight on hand	40 sec	Left Fall	40 sec
Backward fall from seated positions	40 sec	Grabbing while falling	40 sec

Table 1 describes a list of ADLs and the duration it was performed, and similarly Table 2 describes the list of Fall activities and their durations, respectively. All the activities listed in Table 1 and 2 have been performed by the 41 volunteers and each activity was performed for 5 trials by every volunteer. The volunteer statistics from whom the data was collected is summarised in Table 3

TABLE III: Summary of the volunteer statistics

Sr no.	Parameter	Values and Nos
1	Gender	Male = 27 Female = 14
2	Age-range	20-30 years = 29 30-40 years = 6 >40 years = 6
3	Weight-range	50 Kg – 65 Kg = 21 65 Kg – 80 Kg = 16 80 Kg – 100 Kg = 3 100 Kg – 120 Kg = 1
4	Height Range	5ft – 5ft 5in = 23 5ft 5in – 6ft = 16 >6ft = 2
5	Health Issues	No. of subjects with health issues = 17 No. of subjects without health issues = 24 **Health Conditions of subjects:** Sinus Tachycardia, High Blood Pressure, Overweight, Folic acid allergy, Obese, Thyroid, Hypochondria, extreme anxiety Low Blood Pressure, Prostrate, Sinusitis and Genetic Diabetes

While developing Machine Learning algorithms, only a subset of the dataset is useful for building the model. The rest of the data may either be irrelevant or redundant. A feature is considered as an attribute that impacts the solving of a problem, and feature selections are all about selecting the most important features to train the ML Model. Feature Engineering is a vital part of Machine Learning. This is mainly made up of two processes :

- Feature Extraction [20]
- Feature Selection [21].

In the case of feature extraction, a new set of features is created from the existing raw data. Our dataset which we had collected from 41 volunteers has closed to a million datapoints per volunteer (which included Accelerometer, magnetometer and gyroscope values). Out of the million datapoints, 112 features per volunteer were extracted.

The statistical features that were extracted include (i). median (ii). skew (iii). kurtosis (iv).max (v). min (vi). standard deviation (vii). variance.

Feature Selection reduces the number of input variables (raw data) by using only relevant raw data to reduce the overfitting of the model. Researchers either use feature extraction or feature selection. Both feature extraction and feature selection were performed by first extracting the statistical features and then selecting the relevant data from the statistical features extracted. The overall number of features extracted was 112 feature per user per activity per trial, we had 5 trials each. Once feature extraction was performed data pruning was done to select features in order to reduce the amount of data required to train the various ML/DL models.

Dataset pruning removes suboptimal tuples and redundant data to improve the performance of the ML model. This is specifically used with ensemble techniques. Pruning reduces the complexity of the final model, removing any overfitting while reducing the latencies when the trained model is run on real-time data. The ML/DL Algorithms, especially ensemble techniques such as XGBoost, AdaBoost, and our proprietary ensemble algorithms, were run on a constrained device, Qualcomm Snapdragon 820c. In that case, we require methods to drastically reduce the dataset dimensionality without losing any information. So, we need to identify the features that have the maximum impact on the algorithm's performance. There are several methods available for feature selection. Some of these methods are also used for validating the predictions. Of these, we used four different methods for the Ensemble Algorithms.

A. Method 1: Drop Column

We ran the various ML Algorithms by removing one feature at a time, and then a combination of features was tried. We repeated this until we obtained an ideal number of features.

B. Method 2: Shapely Additive exPlantation [23][24]

Shap assigns an importance value for each feature based on every predication. Using these importance values, SHAP thereby identifies the features that influence the predictions. A combination of Local Interpretable Model agnostic explanations (LIME) [25], Layer-wise Relevance Propagation [26] and Deep Lift [27].

In this paper, we have used model-agnostic kernel SHAP for feature selection. We chose kernel SHAP as it works well for a few number of inputs. Moreover, the kernel SHAP method requires fewer evaluations of the original model to obtain similar accuracy. SHAP explains an instance's prediction by computing each feature's contribution in making the prediction. In kernel SHAP, the Shapely valued exPlanation is represented as an additive feature attribution method, a linear model. This connects the LIME and the Shapely Values.

Thereby, the Shapely Values are the only solutions that satisfy the properties of :

(a) for a specific input (x), local accuracy requires the exPlanation model to atleast match the output (f) for a simplified input (x).

(b) local accuracy when approximating the original model

(c) Consistency – states that if a model changes so that some input contribution increases or stays the same regardless of the other inputs, that input attribution should not decrease.

(d) Missingness – If the simplified input represents feature presence, the missingness requires features missing from the original input to have no impact on the prediction.

We selected kernel shap as it is model agnostic. Kernel SHAP identifies the class of additive feature's importance, and hence, these important values are used for pruning the feature-extracted dataset. Kernel SHAP estimates, for instance, x, the contribution of each feature in making the prediction. We have used this for ranking the features and selecting the top-ranked features.

C. Method 3: Correlation combined with RELIEFF

We initially used cross-correlation to remove redundant features from the extracted dataset and then applied RELIEFF [28] to remove low-impact features. RELIEFF is a modification of RELIEF [29] that can remove irrelevant features by estimating the relevance of all features. RELIEFF improves on the original algorithm by estimating the probabilities more reliably and can also be used in the case of small and incomplete datasets.

RELIEF estimates how parameters compare to the instances that are close to it. RELIEF searches for two neighbours –

- From the exact prediction (class) called the nearest hit
- From a different prediction called the nearest miss.

RELIEF then uses Manhattan's distance to find the difference between the two parameters. A weight is assigned to the parameters based on the difference. RELIEF uses impurity functions. The correlation between a prediction and a parameter constitutes the impurity function, where the impurity function disregards the context of the parameters that have no correlation with the prediction.

RELIEFF is a modification of relief that uses standard linear correlation coefficients to estimate the contribution of each parameter. For parameters that are conditionally independent, as the number of nearest neighbour increase, the quality of the estimate increases monotonically. In case of dependent attributes, the estimation quality decreases monotonically. RELIEFF works well with both dependent and independent attributes and hence is a good method of pruning for the dataset we have collected, as the dataset is small and noisy. Any ML algorithm should try to discover regularity in the data, which is done by RELIEFF.

RELIEFF also reduces the complexity where RELIEF takes every single hit and single miss and finds the difference. RELIEFF takes (k) nearest hits and (k) nearest misses and calculates the average distance. RELIEFF uses Euclidean distances, although the paper clearly does not explain why Euclidean distance was used instead of Manhattan's distance. The results and discussion section describe the result of pruning using RELIEFF. There are several other pruning methods available, such as Firefly, STIR [30], etc, that are more suited for large datasets, images, and Deep Learning methods where the prediction is not binary.

This way, our work is completely novel compared to other works based on data pruning.

D. Method 4: MIT Lottery Ticket Hypothesis

Neural network pruning techniques can reduce computation and storage requirements without reducing accuracy. The MIT lottery ticket hypothesis states, "Randomly initialized dense network consists of a subnetwork that is initialized such that – when trained in isolation, it can match the test accuracy of the original network after training for at the most same number of iterations. ". We have applied the MIT lottery ticket hypothesis to MLP, CNN and RNN models and have found individual improvement in performance. We have applied the same to the ensemble algorithm described in section 4. Generally, in iterative pruning, the unpruned weights are reset to their original values before the next iteration; we have not done so. So, we have modified the algorithm to obtain the best possible results [31].

IV. ENSEMBLE ALGORITHM

The ensemble algorithm developed is based on voting. We have used a weighted combination of A) Stack(A) [32], B) Multi-Layer Perceptron (MLP) and C) Convolution Neural Network (CNN). Individually, the accuracies and Specificity of Stack(A), MLP and CNN were as follows:

- Stack A
  - Accuracy = 97.62%
  - Specificity = 98.15%
- CNN
  - Accuracy = 94.64%
  - Specificity = 89.83%
- MLP
  - Accuracy = 96.43%
  - Specificity = 94.64%

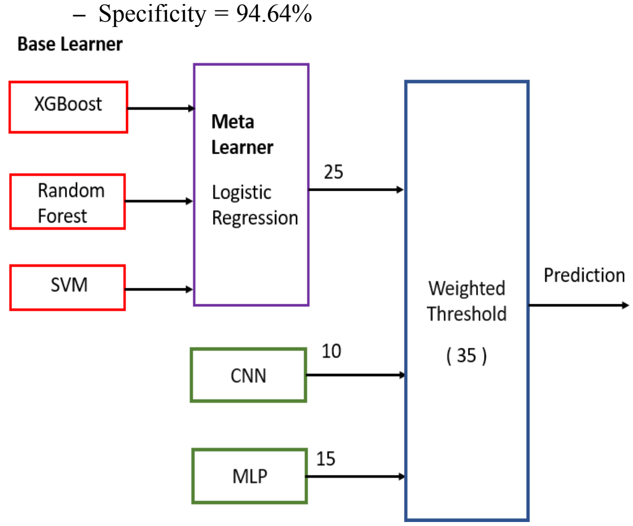


Fig. 3: Block Diagram of VWE(D)

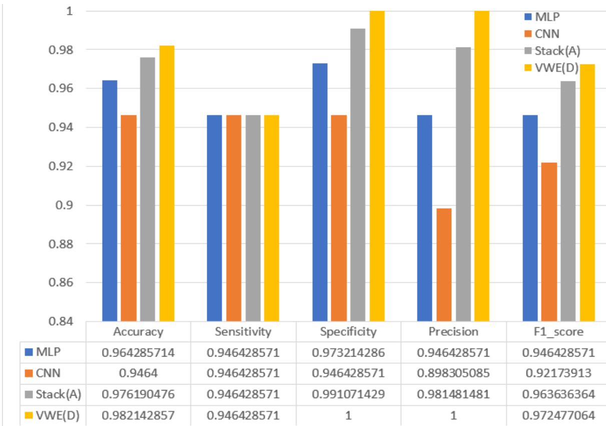
We ran multiple iterations of varying weights to find the ideal weights, which produced an accuracy of 98.21% with 100% specificity, meaning no ADL activities are going undetected. The precision obtained was 94.64%, and the F1 Score was 97.25%, which falls under the category of outstanding. Based on various literature studied, this is, to the best of our knowledge, the algorithm that gives the most accurate results across varied user dynamics while the algorithm runs on a constrained SoC.

The block diagram of the ensemble algorithm and the equation used is shown in Figure 3.

This is a Multi-Layered Ensemble Algorithm as Stack(A) already uses the stacking ensemble method with the primary layer made of Support Vector Machine (SVM), extreme Gradient Boosting (XGB), Random Forest (RF) with Logistic Regression (LR) as the Meta Learner where XGB and RF are already ensemble algorithms. Our logic behind using these algorithms is described in [32] .

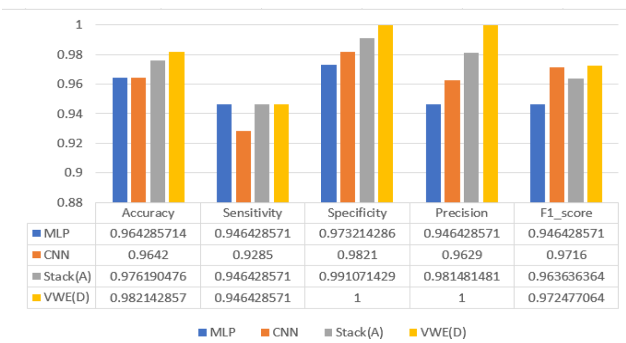
These weights of 25 for Stack(A), 10 for CNN and 15 for MLP were decided after trying multiple combinations. As the number of combinations we tried was very high, it was impossible to show the accuracy of all the combinations.

V. Results and Discussion



	Accuracy	Sensitivity	Specificity	Precision	F1_score
MLP	0.964285714	0.946428571	0.973214286	0.946428571	0.946428571
CNN	0.9464	0.946428571	0.946428571	0.898305085	0.92173913
Stack(A)	0.976190476	0.946428571	0.991071429	0.981481481	0.963636364
VWE(D)	0.982142857	0.946428571	1	1	0.972477064

Fig. 4: Performance metrics without modified MIT lottery hypothesis



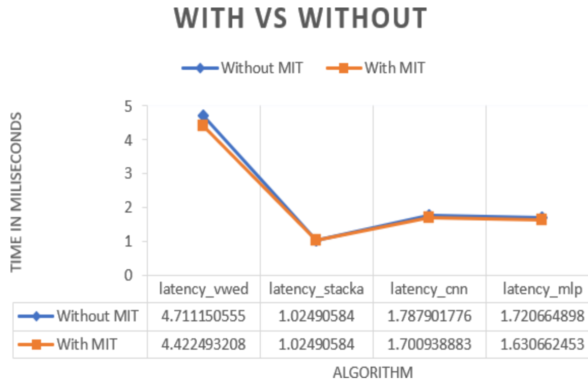
	Accuracy	Sensitivity	Specificity	Precision	F1_score
MLP	0.964285714	0.946428571	0.973214286	0.946428571	0.946428571
CNN	0.9642	0.9285	0.9821	0.9629	0.9716
Stack(A)	0.976190476	0.946428571	0.991071429	0.981481481	0.963636364
VWE(D)	0.982142857	0.946428571	1	1	0.972477064

Fig. 5: Performance metrics modified MIT lottery hypothesis

Figure 4 and Figure 5 give the various parameters (accuracy, sensitivity, specificity, precision and F1 Score) with and with- out the use of the Modified MIT Lottery hypothesis. While the accuracy of MLP remains the same, there is a slight increase in the accuracy for CNN (increases from 94.64% to 96.42%), and there is an increase in specificity while there is a drop in sensitivity. There is a large increase in precision (from 89.83% to 96.29%). There is also more than 2% increase in the F1 Score. We can conclude that by using the Modified MIT Lottery, CNN produces better results at lower latencies, while there is no drop in accuracy in the case of MLP. When VWE(D) is applied, the parameters are the same for with Modified MIT and without Modified MIT lottery ticket.

This is due to the applied. Stack(A) and MLP together have higher accuracy, however, the accuracy remains unaffected when using CNN.

The latencies are shown in Figure 6 for with and without the use of Modified MIT Lottery Ticket. The latency drops from 4.711ms to 4.422ms when Modified MIT Lottery is used.



	latency_vwed	latency_stacka	latency_cnn	latency_mlp
Without MIT	4.711150555	1.02490584	1.787901776	1.720664898
With MIT	4.422493208	1.02490584	1.700938883	1.630662453

Fig. 6: Latency of VWE(D) algorithm with and without MIT Lottery hypothesis

This allows us to sample at higher frequencies if required. But in either case, the sampling frequency of 20 Hz between two adjacent samples is much higher than the data processing rate. The diagram does not include the time for feature extraction, but the time taken for feature extraction will not add much to the existing latency.

Though the increase in latency is about 3.4 milliseconds with only an increase of 0.6% in accuracy, but the precision increases by almost 2%, and the specificity also increases to 100%. The increase in latency is acceptable since our data sampling rate is 500 milliseconds. Even if feature extraction and data gathering time are included, the prediction would be done in less than ten milliseconds, which is much less than 500 milliseconds. Hence, we can afford to increase the latency even though the accuracy increases only by a small value, but there is a huge impact on specificity and precision, both being close to 100%. Only three falls remained undetected, and all of them involved falling from a lying down position on the bed. Since the bed was about 10 centimetres from the ground, the change in magnetometer value would have been very less and could be easily confused with ”turning on the side, while lying on the back” ADL activity. And again, these falls were missed in the case of volunteers who were well above the average weight. This is the only soft fall that is going undetected by the system.

VI. Conclusions and Future Directions

Based on the results, we can see that we have reached a peak value of specificity with 98.3% accuracy. Very little further scope for improvements in terms of algorithms exists. We have also built a low-cost custom wrist-worn end device that is capable of collecting data, extracting the features and running the ensemble algorithm, providing the prediction before the next set of samples are collected. The prototype of the custom end device is ready. In order to further study the performance of the end device, we are currently digital twinning the end device so that we can observe every activity performed by the user remotely if required. We are also reverse engineering the end device to produce synthetic data that can be used for further training algorithms and generating varied data sets.

REFERENCES

[1] D. o. E. United Nations, W. P. A. Social Affairs, World population ageing 2020 highlights: Living arrangements of older persons, 2020

[2] Anita Ramachandran, Anupama Karuppiah, ”A Survey on Recent Advances in Wearable Fall Detection Systems”, BioMed Research International, vol. 2020, Article ID 2167160, 17 pages, 2020. https://doi.org/10.1155/2020/2167160

[3] Gusev, M. What makes Dew computing more than Edge computing for Internet of Things. In Proceedings of the 2021 IEEE 45th Annual Computers, Software, and Applications Conference (COMPSAC), Madrid, Spain, 12–16 July 2021; IEEE: Piscataway, NJ, USA, 2021; pp. 1795–1800.

[4] S. Ray, ”A Quick Review of Machine Learning Algorithms,” 2019 International Conference on Machine Learning, Big Data, Cloud and Parallel Computing (COMITCon), Faridabad, India, 2019, pp. 35-39, doi: 10.1109/COMITCon.2019.8862451.

[5] Sarker, I.H. Machine Learning: Algorithms, Real-World Applications and Research Directions. SN COMPUT. SCI. 2, 160 (2021). https://doi.org/10.1007/s42979-021-00592-x

[6] Yang, Li and Abdallah Shami. “On Hyperparameter Optimization of Machine Learning Algorithms: Theory and Practice.” ArXiv abs/2007.15745 (2020): n. pag.

[7] Ngiam KY, Khor IW. Big data and machine learning algorithms for health-care delivery. Lancet Oncol. 2019 May;20(5):e262-e273. doi: 10.1016/S1470-2045(19)30149-4. Erratum in: Lancet Oncol. 2019 Jun;20(6):293. PMID: 31044724

[8] Alloghani, M., Al Jumeily, D., Mustafina, J., Hussain, A., Aljaaf, A.J. (2020). A Systematic Review on Supervised and Unsupervised Machine Learning Algorithms for Data Science. In: Berry, M., Mohamed, A., Yap, B. (eds) Supervised and Unsupervised Learning for Data Science. Unsupervised and Semi-Supervised Learning. Springer, Cham. https://doi.org/10.1007/978-3-030-22475-2[1]

[9] Burrell, J. (2016). How the machine ‘thinks’: Understanding opacity in machine learning algorithms. Big Data & Society, 3(1). https://doi.org/10.1177/2053951715622512

[10] Nandi, P.; Anupama, K.R.; Agarwal, H.; Paliwal, S.; Patel, K.; Bang, V.; Bharat, M.; Guru, M.V. Impact of Varying User Characteristics on The Accuracy of Fall Prediction Algorithms Using Custom Wrist-Worn End Device. Preprints 2023, 2023050917. https://doi.org/10.20944/preprints202305.0917.v2

[11] Qualcomm, Snapdragon 820c development board, 2018. [Accessed 31- Mar-2023]

[12] MAX3010 Heart-Rate and Pulse Oxymeter sensor Datasheet, https://pdf1.alldatasheet.com/datasheetpdf/view859400 / MAXIM/ MAX30102. html , 2018. [Accessed 31- Mar-2023]

[13] “LSM6DSO16ISTR,” *estore.st.com*. Accelerometer Datasheet https://estore.st.com/en/lsm6dso16istr-cpn.html?icmp= tt31374_gl_lnkon_mar2023 (accessed Mar. 21, 2023).

[14] “LIS2MDLTR,” *estore.st.com*. Magnetometer Sensor Datahseet https://estore.st.com/en/lis2mdltr-cpn.html (accessed Mar. 21, 2023)

[15] Vavoulas, G., Pediaditis, M., Chatzaki, C., Spanakis, E. & Tsiknakis,M. The MobiFall Dataset:: Fall Detection and Classification with a Smartphone. *International Journal Of Monitoring And Surveillance Technologies Research*. **2** pp. 44-56 (2016,7)

[16] Sucerquia, A., Lo´pez, J. & Vargas-Bonilla, J. SisFall: A Fall and Movement Dataset. *Sensors*. **17** (2017), https://www.mdpi.com/1424-8220/17/1/198

[17] SmartFall Dataset https://userweb.cs.txstate.edu/ hn12/data/ SmartFall- DataSet/SmartFall [Accessed 31-Mar-2023]

[18] SmartWatch Dataset https://userweb.cs.txstate.edu/ hn12/data/ Smart-FallDataSet/SmartWatch [Accessed 31-Mar-2023]

[19] Notch Dataset https://userweb.cs.txstate.edu/ hn12/data/ SmartFall- DataSet/Notch [Accessed 31-Mar-2023]

[20] Elena Escobar-Linero, Francisco Luna-Perejo´n, Luis Mun˜oz-Saavedra, Jose´ Luis Sevillano, Manuel Dom´ınguez-Morales, On the feature extrac- tion process in machine learning. An experimental study about guided versus non-guided process in falling detection systems, Engineering Applications of Artificial Intelligence, Volume 114, 2022, 105170, ISSN 0952-1976, https://doi.org/10.1016/j.engappai.2022.105170.

[21] B. Venkatesh and J. Anuradha. 2019. A Review of Feature Selection and Its Methods. Cybern. Inf. Technol. 19, 1 (Mar 2019), 3–26. https://doi.org/10.2478/cait-2019-0001

[22] Venkatesh,B. & Anuradha,J.(2019).A Review of Feature Selection and Its Methods. Cybernetics and Information Technologies,19(1) 3-26. https://doi.org/10.2478/cait-2019-0001

[23] Lundberg, S.M., & Lee, S. (2017). A Unified Approach to Interpreting Model Predictions. ArXiv, abs/1705.07874.

[24] Tripathi, S., Hemachandra, N., & Trivedi, P. (2020). Interpretable feature subset selection: A Shapley value based approach. 2020 IEEE International Conference on Big Data (Big Data), 5463-5472.

[25] Gramegna A, Giudici P. SHAP and LIME: An Evaluation of Discrimina- tive Power in Credit Risk. Front Artif Intell. 2021 Sep 17;4:752558. doi: 10.3389/frai.2021.752558. PMID: 34604738; PMCID: PMC8484963.

[26] Chen H, Lundberg SM, Lee SI. Explaining a series of models by propagating Shapley values. Nat Commun. 2022 Aug 3;13(1):4512. doi: 10.1038/s41467-022-31384-3. PMID: 35922410; PMCID: PMC9349278

[27] Ullah I, Rios A, Gala V, Mckeever S. Explaining Deep Learning Models for Tabular Data Using Layer-Wise Relevance Propagation. Applied Sciences. 2022; 12(1):136. https://doi.org/10.3390/app12010136

[28] Kononenko, I., Sˇimec, E. & Robnik-Sˇikonja, M. Overcoming the Myopia of Inductive Learning Algorithms with RELIEFF. Applied Intelligence 7, 39–55 (1997). https://doi.org/10.1023/A:1008280620621

[29] Urbanowicz, R. J., Meeker, M., La Cava, W., Olson, R. S., & Moore, J. H. (2018). Relief-based feature selection: Introduc- tion and review. Journal of biomedical informatics, 85, 189–203. https://doi.org/10.1016/j.jbi.2018.07.014

[30] Yong Zhang, Xian-fang Song, and Dun-wei Gong. 2017. A return-cost- based binary firefly algorithm for feature selection. Inf. Sci. 418, C (December 2017), 561–574. https://doi.org/10.1016/j.ins.2017.08.047

[31] J. Frankle and M. Carbin, “The Lottery Ticket Hypothesis: Find- ing Sparse, Trainable Neural Networks,” 2019. [Online]. Available: https://openreview.net/forum?id=rJl-b3RcF7

[32] Nandi, P.; Anupama, K.; Agarwal, H.; Jain, A.; Paliwal, S.; Jakhar, R.; Rajkhowa, R.; Guruprasanna, P.B. Application of Fea- ture Engineering and Ensemble Techniques on Medical End- devices for Geriatric Fall Detection. Preprints 2023, 2023060804. https://doi.org/10.20944/preprints202306.0804.v1