Detection of Situational Context From Minimal Sensor Modality of A Smartphone Using Machine Learning Algorithm

Nabonita Mitra, *Student Member, IEEE* and Bashir I. Morshed, *Senior Member, IEEE*
Department of Computer Science, Texas Tech University, Lubbock, TX, USA
nmitra@ttu.edu and bmorshed@ttu.edu

2023 IEEE International Conference on Electro Information Technology (eIT) | 978-1-6654-9376-5/23/$31.00 ©2023 IEEE | DOI: 10.1109/EIT57321.2023.10187382

***Abstract*—Early detection and continuous monitoring can help reduce the complexity of treatment and recovery. For this purpose, many modern technologies are being used like smart wearable devices to make the diagnosis of different types of human diseases and automated tutoring systems. There is a vast improvement in the sector of human healthcare and education delivery using artificial intelligence (AI). For these AI algorithms, there can be high error rates if situational contexts are ignored. Currently, there is no automated approach to detect situational context. In this work, we propose a novel approach to automatically detect situational context with a smartphone context detection app using AI from minimal sensor modality. We begin the process by converting a few sensor data from the smartphone app to a multitude of axes, then determine situational context from these axes by using a machine learning algorithm. At first, we evaluated *k-means* algorithm performance on the converted data and grouped them into different clusters according to the contexts. However, the *k-means* algorithm has many challenges that negatively affect its clustering performance. For this reason, to automatically detect the situational contexts more accurately we have performed different machine learning (ML) algorithms to differentiate their characteristic parameters and attributes. To train and test ML models, 145 features were extracted from the dataset. In our case, we have used a dataset with 53,679 distinct values to evaluate the performance of different algorithms in detecting five situational contexts of the users. Experimental result shows that the accuracy of the Support Vector Machine, Random Forest, Artificial Neural Network, and Decision Tree Classifiers are 95%, 99%, 97%, and 98% respectively. The most effective classifier overall is Random Forest. This preliminary work shows the feasibility of detecting situational context automatically from a few sensor data collected from the smartphone app by converting the sensor data to multiple axes and applying a machine learning algorithm.**

Keywords: *k-means* algorithm, machine learning, situational context detection, smartphone app

I. Introduction

The advancements in modern technology have brought about remarkable changes in several aspects of our daily life, making it an essential part of our existence. With the progress made in health monitoring technology, the future of smart devices looks promising, as they will be capable of not only observing their environment but also keeping track of crucial vital signs such as heart rate and breathing rate [1]. The accurate measurement of a patient's condition is possible with the availability of sensors in wearable devices, enabling the continuous monitoring of physiological changes and the progress of treatments. The interpretation of a user's physical situation by an automated system could alert the physician about the patient's condition; however, data dependency poses a significant pitfall as data can sometimes be misleading. Utilizing AI without context-aware systems can result in high error rates and erode users' trust. Therefore, the incorporation of context detection would significantly reduce the error rate in such cases.

So far, wearable technology has been mainly used for monitoring a few health-related parameters [2]. To address this issue, we have conducted research on how we can use wearable sensors to gather data without adding unnecessary complexity to people's lives. One potential solution is to use context-aware systems with smartphones, which are intelligent systems that can suggest adaptive service choices based on the user's current context [3]. One of the main benefits of smartphones is that they are portable and always with the patients, making it easy to collect data on a continuous basis. Situational context refers to the context in which a patient is located or the situation they are in. For example, if a patient's heart rate suddenly spikes while they are driving, this could be an indication of a heart attack or other cardiac event. By using situational context, the smartphone can alert the patient or their healthcare provider to take appropriate action. To make sense of this data, we can apply various machine learning (ML) models as artificial neural networks and Support Vector Machines (SVMs) in classification tasks.

The objective of this research is to analyze the potential of a smartphone app designed for situational context detection. The study aims to determine the usefulness of using sensor data to identify context and subsequently convert the collected data into multiple-axes data to enhance the precision of detecting situational context. We performed the *k-means* algorithm to cluster the converted data according to their contexts. To detect usual patterns and make predictions about the user's situational context, we employed different machine learning models. We compared the classification accuracy results of these models to determine which one is better at detecting contexts. By analyzing the situational context and the data collected from these axes, we can develop a procedure that can assist in comprehending the patient's condition.

978-1-6654-9376-5/23/$31.00 ©2023 IEEE

II. Data Collection and Pre-processing

A. Smartphone Context Detection App

This work involves the use of smartphone sensors to collect important data from the human body, which can then be converted into multiple axes data. For instance, we can measure a person's heart rate using pulse-oximetry and their speed using the GPS function on a smartphone. The collection of these axes data from smartphone sensors is depicted in Fig. 1.

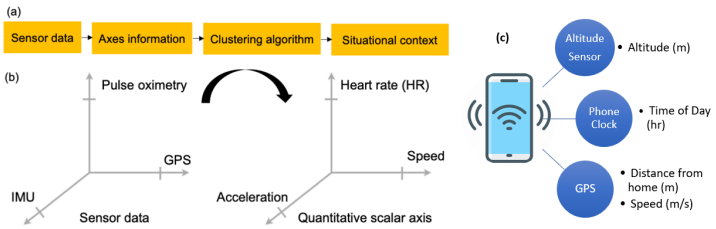

Fig. 1. (a) Steps of analyzing sensor data collection (b) Illustration example of converting 3 sample data to 3 axes data (c) Axes Data collection from smartphone

For our work, we have used a smartphone context detection app that was developed in our lab and can be used on any Android smartphone to collect sensor values. This app uses the built-in sensors of the smartphone and records the activities of the user. Then the collected data from the smartphone was transferred to local computer via CSV file. Different page formats of the smartphone context detection app are shown in Fig. 2.

B. Data Validation

To ensure the accuracy of the collected data from the smartphone app, we followed a protocol that involved a series of steps. We have collected data on 5 situational contexts of the users. The contexts are the sitting, walking, running, driving, and sleeping activities of the users. For sitting, sleeping, and driving positions we followed the same steps which are:

1) Start the smartphone context detection app and select the sensors to collect the data.
2) Keep the smartphone in the user's pocket or hand and sit for 60 seconds steadily.
3) After 60 seconds, the app will be stopped and the collected sensor data will be saved in CSV format.

For walking and running positions we followed some more steps which are:

1) Start the smartphone context detection app and select the sensors to collect the data.
2) Keep the smartphone in the user's pocket or hand and run or walk for 20 seconds.
3) Then rest or stand steadily for 10 seconds.
4) Again, run or walk for 10 seconds.
5) After that, rest or stand steadily for 10 seconds again.
6) Again, run or walk for 10 seconds. That's how we collected all together 60 seconds of running or walking data.

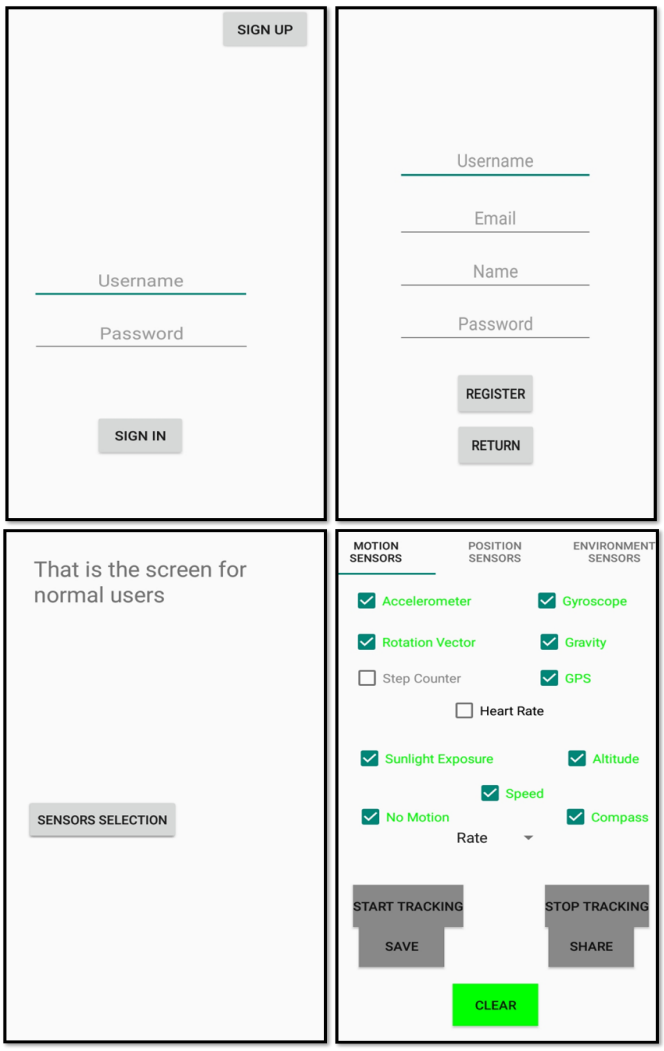

Fig. 2. Different pages of the Smartphone Context Detection App

7) After 60 seconds, the app will be stopped and the collected sensor data will be saved in CSV format.

A series of sequential steps have been formulated for each position, which are displayed on a PowerPoint slide. For our work, we have collected the speed, time, altitude, GPS, and accelerometer sensor values from the smartphone context detection app.

C. Feature Extraction

Extracting features from raw data is a crucial step in data analysis. Our study involved collecting 50 sets of data in various user situations. Once we acquired the sensor data, we processed them to extract a range of parameters to analyze the situational context. In this work, we extracted multiple features from all our collected datasets. Some of the extracted features are outlined below:

1) **Approximate Entropy:** It is a statistical method used to measure the complexity of time series data by quantifying the degree of regularity or irregularity in the data [4].

2) **Kurtosis:** Kurtosis indicates whether the distribution is flatter or more peaked than the normal distribution [5].

$$Kurtosis = 1/n \sum_{i=1}^{N} \frac{\sum_{i=1}^{N} (X_i - X)^4}{a4}$$

Where X, a, and n are the sample mean, the sample standard deviation, and the sample size, respectively.

3) **Mean of absolute deviation:** It is a statistical measure that calculates the average of the absolute differences between each data point in a dataset and the mean of that dataset. It can be defined as [6]

$$Mean\ of\ absolute\ deviation = 1/n \sum_{i=1}^{N} \lfloor X_{i+1} - X_i \rfloor$$

Apart from the above-mentioned features, we have also extracted several other features, which are enlisted in Table I.

TABLE I
KEY FEATURE LIST

Features	
0_Absolute sum of changes	0_Interquartile range
0_Autocorrelation	0_LPCC_0
0_Centroid	0_Max power spectrum
0_ECDF Percentile Count_0	0_Maximum frequency
0_ECDF Percentile Count_1	0_Mean
0_ECDF Percentile_0	0_Median
0_ECDF Percentile_1	0_Median absolute deviation
0_ECDF_0-9	0_Negative turning points
0_Entropy	0_Neighbourhood peaks
0_FFT mean coefficient_0	0_Peak to peak distance
0_FFT mean coefficient_1-10	0_Root mean square

III. METHODS

Our work involves working with sensor data to extract relevant information and convert it into multiple axes data. To ensure accurate clustering and machine learning, we need a substantial dataset categorized according to various axes and contexts. This dataset will serve as the foundation for our analysis, enabling us to identify patterns and make predictions based on the situational context of the user.

A. K-means Algorithm

The *K-means* algorithm is an algorithm used for clustering in machine learning and classification tasks. It works by assigning each data point to the cluster with the nearest mean or centroid, where the mean is the arithmetic average of all the data points in that cluster. The algorithm begins by randomly selecting *K* initial centroids, and then iteratively assigns each data point to the nearest centroid and updates the centroid to the mean of the assigned data points. The algorithm stops when the centroids no longer change, or a predetermined maximum number of iterations is reached. The algorithm can be used for a variety of applications, such as image segmentation, customer segmentation, and anomaly detection [7].

The *k-means* algorithm involves several steps. They are

1) The number of desired clusters, denoted by *K*, is chosen.
2) The algorithm randomly initializes *K* centroids by selecting *K* data points from the dataset without replacement.
3) The distance between each data point and each centroid is calculated, and the data points are assigned to the nearest centroid.
4) The distance between each data point and each centroid is calculated, and the data points are assigned to the nearest centroid

B. Support Vector Machine

The SVM algorithm is a machine learning model that identifies a hyperplane in N-dimensional space, which acts as a decision boundary to separate data points based on their attributes. Its goal is to find the hyperplane that has the widest margin between two classes of data points among all possible decision boundaries for classification [8].

The hyperplanes assist distinguish between data points from various classes. The hyperplane's dimension changes depending on how many input features there are; for two features, it is a line, while for three features, it is a 2D plane. However, it becomes difficult to visualize the hyperplane for feature spaces with increasing dimensions.

The support vectors, or data points closest to the hyperplane, are identified by the SVM algorithm, which then makes use of them to maximize the margin between the classes [9]. As a result, a classification approach that can handle both linear and nonlinear data is produced. It has been demonstrated that the SVM algorithm outperforms other classification methods in a variety of domains, including image classification, bioinformatics, and text classification.

C. Random Forest

A Random Forest (RF) is a machine learning algorithm that uses an ensemble of decision trees to make predictions [10]. Each decision tree in the forest produces a prediction, and the class with the highest number of votes is considered as the final prediction. The strength of RF lies in the idea of the wisdom of crowds, where a large number of relatively uncorrelated decision trees can work together as a committee to improve the accuracy of the model [10].

The algorithm's success depends on the decision trees' minimal correlation with one another. The ensemble predictions made by the uncorrelated trees can be more precise than those made by any single tree. The reason for this is that the trees may correct one other's mistakes, and the forest can progress in the right path with the support of the majority vote. The proportion of test data that the model properly classifies can be used to describe the classification accuracy of the RF algorithm. The accuracy is described as

$$Accuracy = \frac{TP+NP}{TP+TN+FP+FN}$$

In the formula: TP represents the correct positive; TN represents correct negative; FP represents the false positive; FN represents false negative.

D. Decision Tree

Classification is very important in various fields, and one of the most popular tools used for these tasks is the Decision Tree (DT). In a DT, each internal node denotes a test on an attribute, and each branch signifies the result of the test. The terminal nodes, also known as leaf nodes, hold the class label. The advantage of using a DT is that it can generate different rules for decision-making, while requiring less computational resources than other methods [11].

When a new input is presented to the classifier, it traverses the decision tree from the root to a leaf node, following the path that satisfies the decision rules based on the input features. The anticipated class label for the input is represented by the leaf node that is reached at the end of the journey. DT is a highly well-known model for classification and prediction problems since it can pinpoint the most crucial areas for classification or prediction.

E. Artificial Neural Network

Artificial neural network (ANN) is a computational model inspired by the structure and function of the human brain. The ANN consists of a large number of interconnected processing nodes that are organized into layers. The input layer receives the data, the output layer produces the output, and the hidden layers process the data in between. The nodes in each layer are connected to the nodes in the next layer by weights, which determine the strength of the connection between the nodes [12].

The ANN is trained by adjusting the weights of the connections between the nodes, using a learning algorithm. During training, the ANN learns to recognize patterns in the input data and to produce the correct output for each input. Once trained, the ANN can be used to classify new data or to make predictions based on the input data. One of the key advantages of ANNs is their ability to learn and generalize from large amounts of data, making them well-suited for applications where traditional rule-based approaches are not effective [13].

IV. Results

Our study focuses on capturing and analyzing human positions using smartphone sensor data. To achieve this, we selected a combination of five axes and five situational contexts that can be observed from a smartphone while being carried by the user. By collecting data from a context detection app, we were able to gather relevant sensor data for each axis and context. These 5 axes and contexts are outlined in detail in Table II.

TABLE II
List of Axes and Situational Contexts

Axes	Context
Speed (m/s)	Rest
GPS	Walk
Accelerometer	Run
Altitude (m)	Drive
Time of Day (hr)	Sleep

A. K-means Algorithm Results

Once we collected the data from the smartphone app, we proceeded to process it. Since our data set was unsupervised, we decided to use the *k-means* algorithm, which is capable of grouping similar data without the need for training. This algorithm allows us to define the number of clusters for our datasets. To optimize the performance of the *k-means* algorithm with our dataset, we fine-tuned its parameters, as shown in Table III. By doing so, we were able to obtain more accurate and reliable results from our clustering analysis.

TABLE III
K-means Algorithm Parameter Optimization

Parameter	Value
'n_clusters'	5
'n_init'	10
'random_state'	0
Algorithm	Auto
'initialization'	*'k-means++'*
'max_iteration'	300

From Fig. 3, we can observe the spatial arrangement of the data, which has been clustered into 5 distinct situational contexts based on various conditions. By utilizing our dataset, we can identify the specific cluster that corresponds to a test subject's position. This approach provides us with a reliable means of discerning a subject's situational context from the test values.

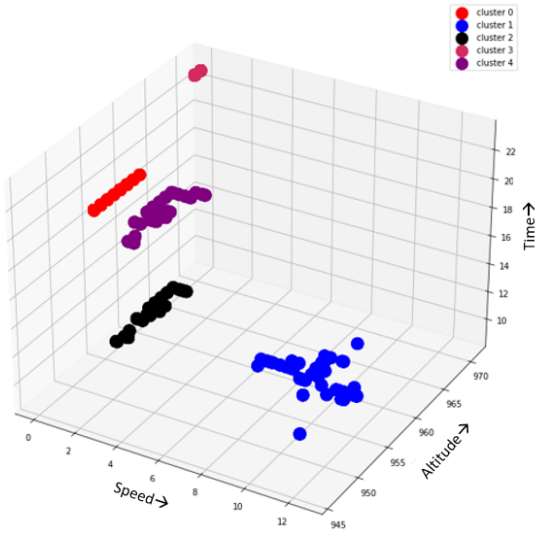

Fig. 3. 3D presentation of the clustering data showing 5 clusters

As we know the *k-means* algorithm is designed for clustering tasks, it is not suitable for multiclass classification tasks where each data point is assigned to one of several predefined

classes. One of the main problems with using *k-means* for the detection of situational context is that it is not designed to handle discrete data. Another issue is that *k-means* assumes that the clusters are spherical and have similar sizes. This assumption may not hold for multiclass datasets, where the classes may have different shapes and sizes, and may overlap with each other. For this reason, to get a more defined detection of situational contexts of our users' different positions we have analyzed the dataset with 4 machine learning models and compared the results to find the best-performing model.

B. Machine Learning Model Results

To get a proper detection and to compare the results, we have analyzed our dataset with 4 machine learning models. They are SVM, RF, DT, and ANN. At first, we trained our dataset with different groups. The 3 dataset split groups are as follows:

In the first approach, the dataset was devided into the training set as 80% and the testing set groups as 20%.

For the second approach, we have split the entire dataset into three groups: training, validation, and testing sets as 70%, 15% and last 15% respectively. This allows us to evaluate our model on a larger testing set and validate the model's performance during the training process.

Lastly, to ensure a robust and accurate evaluation of our model's performance, we have employed the *k*-fold cross-validation estimator, to reduce the variance in performance estimates. With this approach, we divided our dataset into *k* distinct segments and trained and tested our model *k* times, with each segment serving as a testing set once. By setting *k* to 10, we obtained a diverse range of testing sets, which enabled us to evaluate our model's performance in various scenarios.

After training our dataset using the best parameter variation configuration for each of our three dataset distributions, accuracy and performance metrics have been computed. A list of critical parameters of the models is shown in Table IV

TABLE IV
CLASSIFIER KEY PARAMETERS

Classifier	Combination of hyperparameter
SVM	Kernal='rbf'
RF	Number of trees=100
DT	Max_depth=7
ANN	Optimizer='SGD', three layers of neurons

The performance of classifiers is reported in Fig. 4, where the accuracy of different models is presented for different dataset distributions. From our calculated results, we can say that RF performed the best. In Tables V, VI, VII, and VIII we have presented the distribution of the predicted and true class prediction comparison for 4 models for 10-fold cross-validation where the models perform the best.

Table IX presents the precision, recall, and f1-score for different models and distributions. Based on these metrics, the RF classifier shows promising results and can be a suitable choice for our purpose. The performance metrics presentation of 4 models for different dataset distributions are also shown in Figs 5, 6, and 7.

TABLE V
PRESENTATION OF TRUE CLASS AND PREDICTED CLASS FOR SVM MODEL WITH 10-FOLD CROSS VALIDATION SPLIT

Predicted →	Rest	Walk	Drive	Sleep	Run
True ↓					
Rest	866	0	0	235	3
Walk	0	1037	0	0	12
Drive	0	0	1038	0	0
Sleep	275	56	0	755	0
Run	93	48	0	0	984

TABLE VI
PRESENTATION OF TRUE CLASS AND PREDICTED CLASS FOR RF MODEL WITH 10-FOLD CROSS VALIDATION SPLIT

Predicted →	Rest	Walk	Drive	Sleep	Run
True ↓					
Rest	1049	0	0	55	0
Walk	0	1049	0	0	0
Drive	0	0	1038	0	0
Sleep	1	0	0	1085	0
Run	0	0	0	0	1123

V. DISCUSSION

The results of our study show that the RF model outperforms all other models when it comes to classifying five situational contexts using the dataset. With an accuracy rate of up to 99%, the RF model has consistently shown better performance in all of our experiments, whether the dataset was split into an 80/20 split, a 70/15/15 split, or a 10-fold cross-validation split. RFs are highly scalable and can handle large datasets with millions of observations and thousands of features. This

TABLE VII
PRESENTATION OF TRUE CLASS AND PREDICTED CLASS FOR DT MODEL WITH 10-FOLD CROSS VALIDATION SPLIT

Predicted →	Rest	Walk	Drive	Sleep	Run
True ↓					
Rest	1029	0	0	75	0
Walk	0	1049	0	0	0
Drive	0	0	1038	0	0
Sleep	31	0	0	1055	0
Run	17	0	0	0	1106

TABLE VIII
PRESENTATION OF TRUE CLASS AND PREDICTED CLASS FOR ANN MODEL WITH 10-FOLD CROSS VALIDATION SPLIT

Predicted →	Rest	Walk	Drive	Sleep	Run
True ↓					
Rest	1028	0	0	74	2
Walk	0	1046	0	0	3
Drive	0	0	1038	0	0
Sleep	13	0	0	1073	0
Run	0	9	0	0	1114

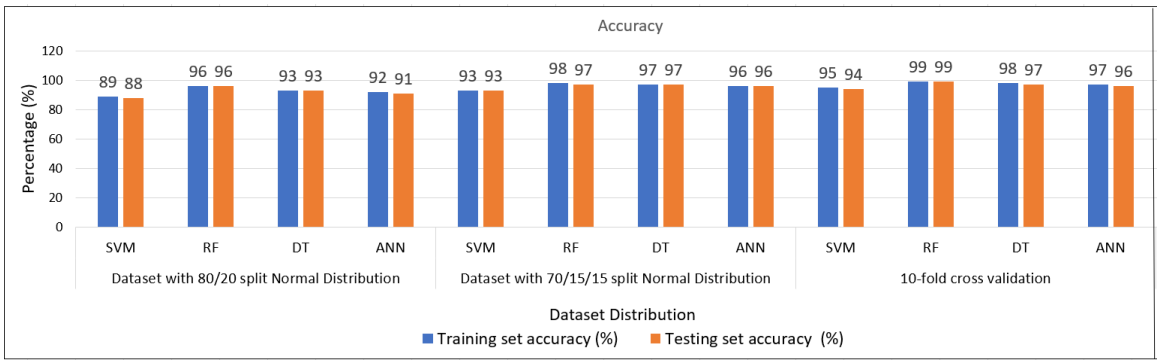

Fig. 4. Accuracy result comparison for different classifier models with 3 dataset distributions

TABLE IX
PRECISION, RECALL AND F1-SCORE PRESENTATION OF DIFFERENT MODELS FOR 10-FOLD CROSS-VALIDATION SPLIT FOR DIFFERENT SITUATIONAL CONTEXTS

	SVM			RF			DT			ANN		
	precision	recall	f1-score	precision	recall	f1-score	precision	recall	f1-score	precision	recall	f1-score
Rest	1	0.93	0.97	1	0.97	0.99	1	0.96	0.98	1	0.93	0.98
Walk	1	1	1	1	1	1	1	1	1	1	1	1
Drive	1	1	1	1	1	1	1	1	1	1	1	1
Sleep	0.94	1	0.97	0.94	1	0.99	0.94	1	0.99	0.98	1	0.97
Run	1	1	1	1	1	1	1	1	1	1	1	1

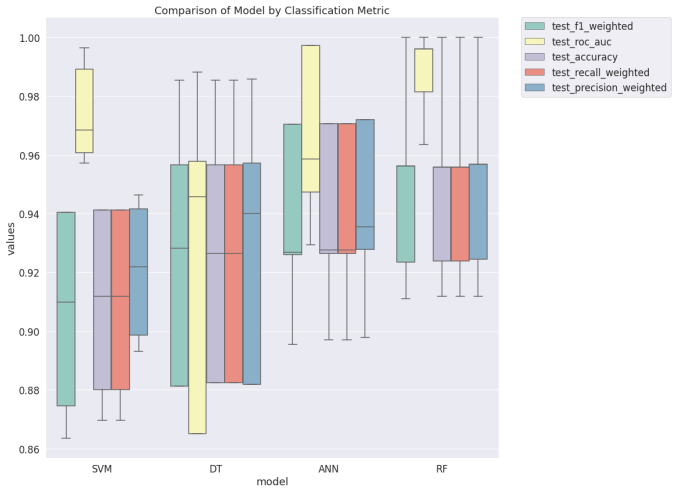

Fig. 5. Performance Metrics presentation of all models for Dataset with 80/20 split

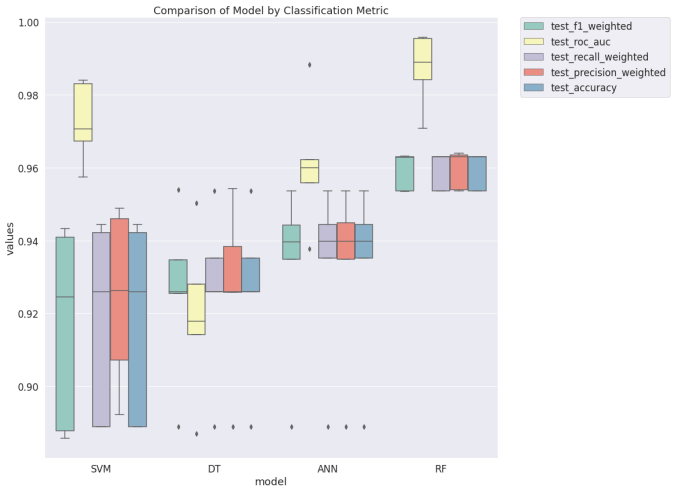

Fig. 6. Performance Metrics presentation of all models for Dataset with 70/15/15 split

algorithm can be parallelized and distributed across multiple processors, making it efficient for big data applications. RFs are relatively easy to interpret and are also known to produce high-accuracy results, especially for complex datasets with high dimensionality. The ensemble nature of the algorithm and the use of bagging and feature randomness techniques help to reduce overfitting and increase generalization performance.

On the other hand, the SVM performs ineffectively with our huge dataset. The SVM is often particularly tricky for expansive datasets, where commotion and exceptions are more likely to be displayed. SVM can take a long time to prepare expansive datasets, particularly on the off chance that the data is not well isolated or the part work is complex. This may make the preparing handle unreasonable for real-time or time-sensitive applications.

VI. CONCLUSION

In this paper, we introduce a novel method for detecting situational context using a limited number of sensors from smartphone. The data collected from a smartphone context detection app was used to develop a classification model for various situational contexts. The data was transformed into

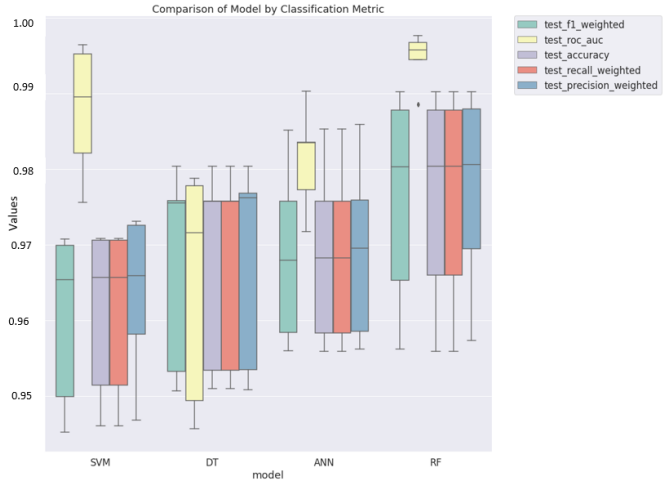

Fig. 7. Performance Metrics presentation of all models for Dataset with 10-fold cross-validation split

multiple axes data, and various machine learning models were performed using real-time data. The results demonstrated that the random forest (RF) model outperformed other models on all three datasets. Our proposed approach has the potential to enhance the performance of automated algorithms for disease detection by optimizing error rates. This approach demonstrates the capability to identify and categorize user situational context from extensive datasets, providing a novel and feasible solution to an important problem. Overall, the presented method holds significant potential for improving the accuracy and reliability of situational context detection systems.

ACKNOWLEDGMENT

This material is based upon work supported by the National Science Foundation under Grant No. 2105766.

REFERENCES

[1] Bashir I. Morshed, Brook Harmon, Md Sabbir Zaman, Md Juber Rahman, Sharmin Afroz and Mamunur Rahman, "Inkjet Printed Fully-Passive Body-Worn Wireless Sensors for Smart and Connected Community (SCC)", J. Low Power Electron. Appl. 2017, 7, 26; doi:10.3390/jlpea7040026 www.mdpi.com/journal/jlpea.

[2] Butte, N. F., Ekelund, U. Westerterp, K. R. "Assessing physical activity using wearable monitors: measures of physical activity", Med. Sci. Sports. Exerc. 44, S5–S12 (2012).

[3] Majumder, S. et al. Smart homes for elderly healthcare—recent advances and research challenges. Sensors 17, 2496 (2017).

[4] S. M. Pincus, "Approximate entropy as a measure of system complexity," Proc. Natl. Acad. Sci. U. S. A., vol. 88, no. 6, pp. 2297–2301, 1991.

[5] C. Orphanidou, T. Bonnici, P. Charlton, D. Clifton, D. Vallance, and L. Tarassenko, "Signal-quality indices for the electrocardiogram and photoplethysmogram: Derivation and applications to wireless monitoring," IEEE J. Biomed. Heal. Informatics, vol. 19, no. 3, pp.832–838, 2015

[6] Santosh K. Divvala, Derek Hoiem, James H. Hays, Alexei A. Efros, Martial Hebert, "An Empirical Study of Context in Object Detection", 978-1-4244-3991-1/09/ ©2009 IEEE.

[7] Jair Cervantes, Farid Garcia-Lamont, Lisbeth Rodríguez-Mazahua, Asdrubal Lopez, A comprehensive survey on support vector machine classification: Applications, challenges and trends, Neurocomputing, https://doi.org/10.1016/j.neucom.2019.10.118.

[8] Ding, Y., Qin, X., He, H.: Parameter Optimizing of Support Vector Machine and Application in Text Classification Computer Simulation (11) (2010) (in Chinese)

[9] Y. Xiao, W. Huang and J. Wang, "A Random Forest Classification Algorithm Based on Dichotomy Rule Fusion," 2020 IEEE 10th International Conference on Electronics Information and Emergency Communication (ICEIEC), Beijing, China, 2020, pp. 182-185, doi: 10.1109/ICEIEC49280.2020.9152236.

[10] Pall Oskar Gislason, Jon Atli Benediktsson, Johannes R. Sveinsson, Random Forests for land cover classification,https://doi.org/10.1016/j.patrec.2005.08.011.

[11] Anuradha and G. Gupta, "A self explanatory review of decision tree classifiers," in International Conference on Recent Advances and Innovations in Engineering (ICRAIE-2014), Jaipur, India, May 2014, pp. 1–7, doi: 10.1109/ICRAIE.2014.6909245

[12] Grossi, Enzo Buscema, Massimo. (2008). Introduction to artificial neural networks. European journal of gastroenterology hepatology. 19. 1046-54. 10.1097/MEG.0b013e3282f198a0.

[13] Shukla, M. Abdelrahman, Mohamed. (2004). Artificial Neural Networks based steady state security analysis of power systems. 36. 266 - 269. 10.1109/SSST.2004.1295661.