Efficient Biometric Gait Authentication by using an accelerometer sensor.

Roberto Vazquez Ferrer, Doctoral Student
Polytechnic University of Puerto Rico
rjvf.007@gmail.com

Jeffrey L. Duffany. PhD
Ana G. Mendez University, jeduffany@uagm.edu

Alfredo Cruz. PhD
Polytechnic University of Puerto Rico
alcruz@pupr.edu

Abstract

Recognizing people by their Biometric-Gait has become more and more popular today. Accelerometer sensors are generally more user-friendly and less invasive. This paper reviews research regarding accelerometer sensors that feature vectors that contribute to Biometric Gait authentication for the person. Smartphones can capture gait patterns through accelerometers and gyroscopes, and innovations in machine learning have started new research paths and applications in gait recognition. The following study will be using the UCI dataset. Also, This dataset contains the motion data of 14 healthy older-aged between 66 and 86 years old. Besides, the following contributions provide a PCA assessment analysis that identifies biometric gait classification based on the XYZ-accelerometer. The results will contribute to the importance can be the use of accelerometer sensors and machine learning to recognize the Biometric-Gait of the person.

Keywords: Machine Learning, Biometric, Gait, Feature Extraction, Support Vector Machine, Principal Component Analysis, Accelerometer.

Introduction

Machine learning has evolved due to the increasing availability of various types of sensor data. Some sensors include smoke detectors, motion sensors, temperature sensors, oximeter sensors, and contact sensors. Hardware is becoming smaller, and sensors are getting cheaper, making sensor data available for various applications, from predictive maintenance to behavior monitoring. One of the most challenging challenges is extracting features from the sensor data because the Machine Learning model suffers overfitting by classification when using the raw dataset without feature extraction.

Biometric Gait is the challenge of classifying a series of accelerometer data logged through dedicated binds or smartphones into defined moving activities. One problem is the large number of observations produced each second and the absence of a straightforward method to connect accelerometer data of well-known moving activities. Also, classical approaches to the problem involve feature vectors consisting of time series based on fixed-size windows and training machine learning models with limited Principal Components Analysis (PCA) analysis.

Attal et al. (2015) review different classification techniques used to recognize human activities from wearable inertial sensor data. Three main steps describe the activity recognition process: sensors' placement, data pre-processing, and data classification. Four supervised classification techniques, namely, k-Nearest Neighbor (k-NN), Support Vector Machines (SVM), Gaussian Mixture Models (GMM), and Random Forest (RF), as well as three unsupervised classification techniques, namely, k-Means, Gaussian

mixture models (GMM) and Hidden Markov Model (HMM), are compared in terms of correct classification rate, F-measure, recall, precision, and specificity. Raw data and extracted features are used separately as inputs of each classifier. The feature selection does perform using a wrapper approach based on the random forest (RF) algorithm. Finally, the results show that the k-NN classifier performs better than other supervised classification algorithms. In contrast, the HMM classifier is the one that gives the best results among unsupervised classification algorithms (Attal et al., 2015).

According to Chen et al. (2021), sensor-based activity recognition and feature extraction are more difficult because of an inter-activity similarity in human activity recognition. Different activities may have similar characteristics (e.g., walking and running). Therefore, it is challenging to uniquely produce distinguishable features to represent activities (Chen et al., 2021).

The goal is to develop a study that researches the benefit of accelerometer sensor devices for biometric gait authentication by using the combination of SVM and PCA. Also, the Support Vector Machine (SVM) is a famous supervised Machine Learning Algorithm that can use for classification and regression tasks by creating a set of hyperplanes and works well in classification problems (Dahanayake et al., 2020). Moreover, the Principal Component Analysis (PCA) helps reduce the dimensionality of extensive datasets vectors and transforms them into orthogonal projection features in lower-dimensional space that contains the most relevant information from the raw datasets. (Dong & Liu, 2018; Raschka & Mirjalili, 2019). In addition, Scikit-learn (Sklearn) is the most valuable and robust library for machine learning in Python. The libraries provide tools for machine learning and statistical modeling, including classification, regression, clustering, and dimensionality reduction (Raschka & Mirjalili, 2019).

Also, the dataset utilized by Shinmoto Torres et al. (2016) from the UCI dataset consists of the motion data of 14 healthy older-aged between 66 and 86 years old who performed approximately written activities using a batteryless, wearable sensor on top of their clothing at sternum level. Data is sparse and noisy due to the use of a passive sensor. The two clinical room settings (S1 and S2) assign to the participants. S1 (Room1) setting uses 4 RFID reader antennas around the room to collect data. In contrast, the S2 (Room2) space setting uses 3 RFID reader antennas (two at ceiling level and one at wall level) to collect motion data. The activities performed were: walking to the chair, sitting on the chair, getting off the chair, walking to bed, lying on the bed, getting off the bed and walking to the door, and Ambulating. The research will focus on walking activities for Biometric Gait authentication for the 14-person classification.

Methodology

The following experiments will use the scientific method to establish the cause-effect relationship among variables.

Experiment one: Consists of determining the windows size data point using raw datasets and SVM. Also, The SVM Recall analysis will create for choice between two classifications derived from Room1 or Room2.

The experiment helps answer:

1. How does SVM help the appropriate window size based on the recall parameter?

Result Experiment:

Figure 1 shows the raw dataset size by subject tag. The minimum size is seven, and the maximum length is 4,638. The window slide shall be less than seven because it represents the minimum size of the subject tag.

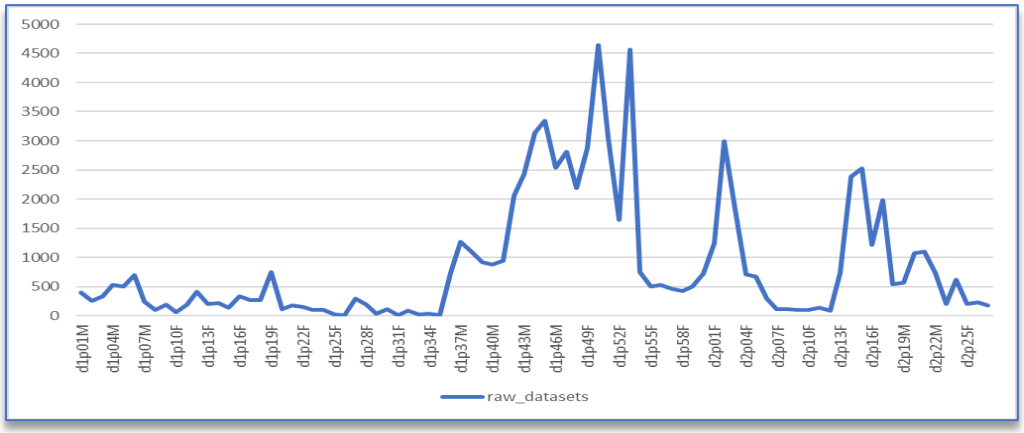


Figure 1: raw dataset size by subject tag

Figure 2 shows the SVM Recall parameter analysis to choose the high average percent. In this case, the winner is the window slide five. Another piece of information is that window slide six tends to overfit tendencies because of the Recall parameter decrement.

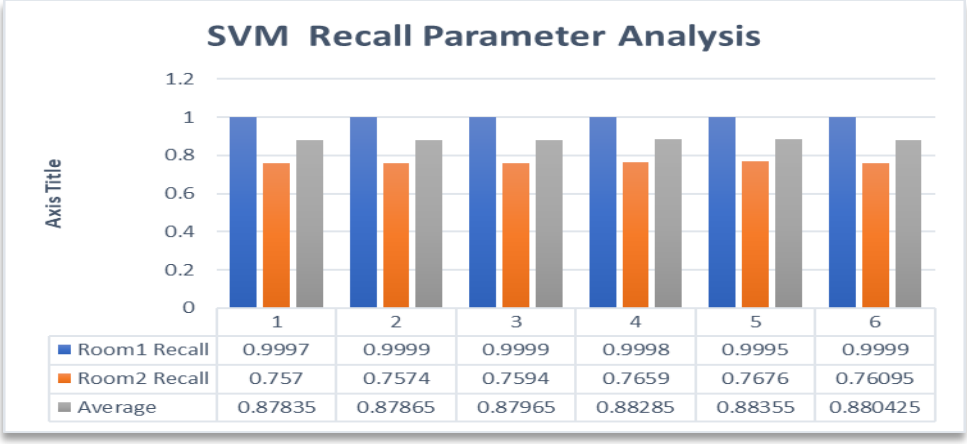


Figure 2: SVM Recall Parameter Analysis

Figure 3 shows the PCA analysis by subject_tag considering the three decimal factors of XY PCA outcome to choose the minimum overlap size. In this case, the winner is the window slide six. However, based on SVM between rooms, the best choice is the window slide five, which consists of overfitting considerations.

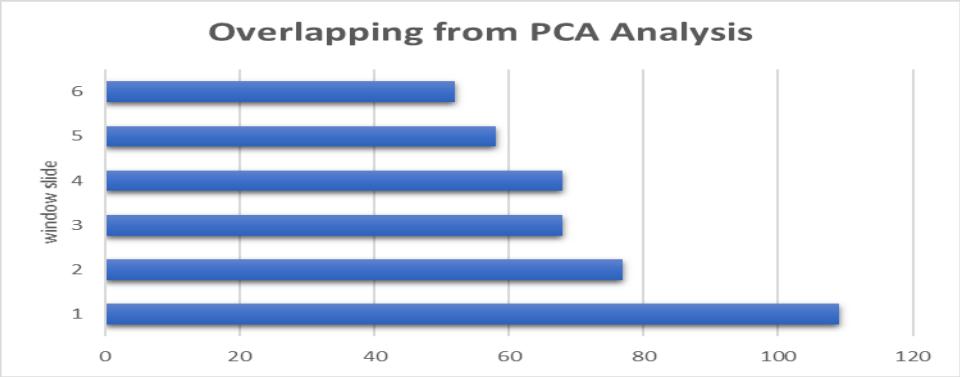
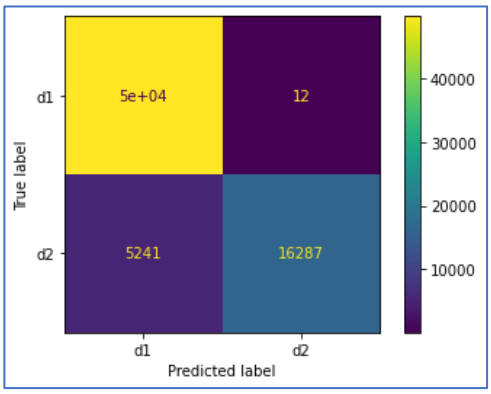
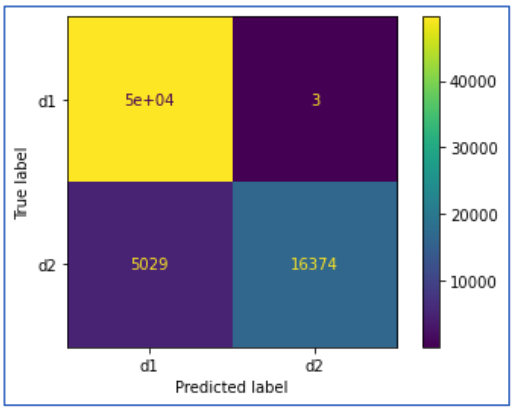


Figure 3: PCA Analysis based on overlapping

Figures 4 (a) & (b) show the before and after window slide selection and the incrementation of the performance observed when classifying between room1 and room2 for Figure (b), which represents the window slide of five.



(a)



(b)

Figure 4: Confusion Matrix Analysis

Experiment two: Consist of defined features for unique identifier baseline for five data components window slide for relative orientation for the sensor. The use of Magnitude can solve the direction of the sensor because, in theory, the magnitude result can be the same result based on the orientation of the devices. Finally, the dataset was split into two tables for room location with the unique key defined with feature extraction and compared to observe non-overlapping. Also, the experiment adds new techniques for the most significant datasets.

The experiment helps answer:

1. How does Magnitude help create a unique identifier between two classifications for entire datasets?
2. How does delta time help create a unique identifier between two classifications for entire datasets?
3. How does Principal Component Analysis (PCA) help identify overfitting-based classification on a two-dimension visualization from raw datasets?
4. How does the sklearn SVM classify the subject tag for biometric gait authentication classification?
5. How does the sklearn k-NN classify the subject tag for biometric gait authentication classification?
6. How does the sklearn Random Forest classify the subject tag for biometric gait authentication classification?
7. How does the sklearn Hash Table classify the subject tag for biometric gait authentication classification?

Result Experiment:

Figure 5 shows the structure of the window slide five. Based on this information, the research explores how to convert the data into a relative form.

	Window Slide				
	1	2	3	4	5
ax	ax1	ax2	ax3	ax4	ax5
ay	ay1	ay2	ay3	ay4	ay5
az	az1	az2	az3	az4	az5
time	time1	time2	time3	time4	time5

Figure 5: window slide five composition

Table 1 shows the feature does use to construct the unique key.

Table 1: Feature Extraction Definition

Feature	Definition
xMag = $\left(\sqrt{\sum_{n=1}^{5} ax_n^2}\right)$	The equation represents the Magnitude for acceleration on the five x-axis data points.
yMag = $\left(\sqrt{\sum_{n=1}^{5} ay_n^2}\right)$	The equation represents the Magnitude for acceleration on the five y-axis data points.
zMag= $\left(\sqrt{\sum_{n=1}^{5} az_n^2}\right)$	The equation represents the Magnitude for acceleration on the five z-axis data points.
timeMag= $\left(\sqrt{\sum_{n=1}^{5} time_n^2}\right)$	The equation represents the Magnitude for acceleration on the five-time data points.
Mag=$\left(\sqrt{ax^2 + ay^2 + az^2}\right)$	The equation represents the Magnitude of each data point of XYZ acceleration.
dt = $(time_n - time_{(n-1)})$	The equation represents the delta time for each interval time data point.
CASE WHEN $(ax_n - ax_{(n-1)})$ <0 **THEN** -1 **ELSE** 1	The equation represents the delta acceleration minus zero of each interval x-axis data point.
CASE WHEN $(ay_n - ay_{(n-1)})$ <0 **THEN** -1 **ELSE** 1	The equation represents the delta acceleration minus zero of each interval y-axis data point.
CASE WHEN $(az_n - az_{(n-1)})$ <0 **THEN** -1 **ELSE** 1	The equation represents the delta acceleration minus zero of each interval z-axis data point.
CASE WHEN $(ax_n - ax_{(n-1)})$!=0 **THEN** -1 **ELSE** 1	The equation represents the delta acceleration not equal to zero of each interval z-axis data point.
CASE WHEN $(ay_n - ay_{(n-1)})$!=0 **THEN** -1 **ELSE** 1	The equation represents the delta acceleration not equal to zero of each interval z-axis data point.

CASE WHEN $(az_n - az_{(n-1)})$!=0 **THEN** -1 **ELSE** 1	The equation represents the delta acceleration not equal to zero of each interval z-axis data point.

Figures 6 show the feature extraction query for window slide five. The intention consists of obtaining the Magnitude and delta time. Also, the feature extraction dataset will perform a PCA analysis for no overlapping do found when converting the result into a unique identity.

```
CREATE TABLE fe_five as
SELECT DISTINCT
       pk_id,
       (time2-time1) as dt1,
       (time3-time2) as dt2,
       (time4-time3) as dt3,
       (time5-time4) as dt4,
       (time5-time1) as dt5,
       (CASE WHEN (ax2-ax1) < 0 THEN -1 ELSE 1 END) AS mx1,
       (CASE WHEN (ax3-ax2) < 0 THEN -1 ELSE 1 END) AS mx2,
       (CASE WHEN (ax4-ax3) < 0 THEN -1 ELSE 1 END) AS mx3,
       (CASE WHEN (ax5-ax4) < 0 THEN -1 ELSE 1 END) AS mx4,
       (CASE WHEN (ax5-ax1) < 0 THEN -1 ELSE 1 END) AS mx5,
       (CASE WHEN (ay2-ay1) < 0 THEN -1 ELSE 1 END) AS my1,
       (CASE WHEN (ay3-ay2) < 0 THEN -1 ELSE 1 END) AS my2,
       (CASE WHEN (ay4-ay3) < 0 THEN -1 ELSE 1 END) AS my3,
       (CASE WHEN (ay5-ay4) < 0 THEN -1 ELSE 1 END) AS my4,
       (CASE WHEN (ay5-ay1) < 0 THEN -1 ELSE 1 END) AS my5,
       (CASE WHEN (az2-az1) < 0 THEN -1 ELSE 1 END) AS mz1,
       (CASE WHEN (az3-az2) < 0 THEN -1 ELSE 1 END) AS mz2,
       (CASE WHEN (az4-az3) < 0 THEN -1 ELSE 1 END) AS mz3,
       (CASE WHEN (az5-az4) < 0 THEN -1 ELSE 1 END) AS mz4,
       (CASE WHEN (az5-az1) < 0 THEN -1 ELSE 1 END) AS mz5,
       (CASE WHEN (ax2-ax1) != 0 THEN -1 ELSE 1 END) AS mtx1,
       (CASE WHEN (ax3-ax2) != 0 THEN -1 ELSE 1 END) AS mtx2,
       (CASE WHEN (ax4-ax3) != 0 THEN -1 ELSE 1 END) AS mtx3,
       (CASE WHEN (ax5-ax4) != 0 THEN -1 ELSE 1 END) AS mtx4,
       (CASE WHEN (ax5-ax1) != 0 THEN -1 ELSE 1 END) AS mtx5,
       (CASE WHEN (ay2-ay1) != 0 THEN -1 ELSE 1 END) AS mty1,
       (CASE WHEN (ay3-ay2) != 0 THEN -1 ELSE 1 END) AS mty2,
       (CASE WHEN (ay4-ay3) != 0 THEN -1 ELSE 1 END) AS mty3,
       (CASE WHEN (ay5-ay4) != 0 THEN -1 ELSE 1 END) AS mty4,
       (CASE WHEN (ay5-ay1) != 0 THEN -1 ELSE 1 END) AS mty5,
       (CASE WHEN (az2-az1) != 0 THEN -1 ELSE 1 END) AS mtz1,
       (CASE WHEN (az3-az2) != 0 THEN -1 ELSE 1 END) AS mtz2,
       (CASE WHEN (az4-az3) != 0 THEN -1 ELSE 1 END) AS mtz3,
       (CASE WHEN (az5-az4) != 0 THEN -1 ELSE 1 END) AS mtz4,
       (CASE WHEN (az5-az1) != 0 THEN -1 ELSE 1 END) AS mtz5,
       sqrt(pow(TIME1,2)+pow(TIME2,2)+pow(TIME3,2)+pow(TIME4,2)+pow(TIME5,2) ) AS timeMag,
       sqrt(pow(ax1,2)+pow(ax2,2)+pow(ax3,2)+pow(ax4,2)+pow(ax5,2) ) AS xMag,
       sqrt(pow(ay1,2)+pow(ay2,2)+pow(ay3,2)+pow(ay4,2)+pow(ay5,2) ) AS yMag,
       sqrt(pow(az1,2)+pow(az2,2)+pow(az3,2)+pow(az4,2)+pow(az5,2) ) AS zMag,
       sqrt(pow(ax1,2)+pow(ay1,2)+pow(az1,2) +pow(TIME1,2) )  as magt1,
       sqrt(pow(ax2,2)+pow(ay2,2)+pow(az2,2) +pow(TIME2,2)  ) as magt2,
       sqrt(pow(ax3,2)+pow(ay3,2)+pow(az3,2) +pow(TIME3,2) )   as magt3,
       sqrt(pow(ax4,2)+pow(ay4,2)+pow(az4,2) +pow(TIME4,2) )   AS magt4,
       sqrt(pow(ax5,2)+pow(ay5,2)+pow(az5,2) +pow(TIME5,2) )   as magt5,
       sqrt(pow(ax1,2)+pow(ay1,2)+pow(az1,2) )  as mag1,
       sqrt(pow(ax2,2)+pow(ay2,2)+pow(az2,2) )as mag2,
       sqrt(pow(ax3,2)+pow(ay3,2)+pow(az3,2) )  as mag3,
       sqrt(pow(ax4,2)+pow(ay4,2)+pow(az4,2) ) AS mag4,
       sqrt(pow(ax5,2)+pow(ay5,2)+pow(az5,2) )  as mag5,
       loc,
       subject_tag
from
dataset_window_slide_5
```

Figure 6: Feature Extraction Table

Figures 7 show the PCA analysis source code in Python. The code search for overlapping depends on the Factor. In this case, the research uses an incremental Factor to observe which case no contains overlapping.

```
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
import sqlite3
from sklearn.preprocessing import StandardScaler
from sklearn.decomposition import PCA


def pcaAnalysis(dataset, factor):
   X = dataset.iloc[:, 1:-1].values
   Y = dataset.iloc[:, dataset.shape[1]-1].values
   ###
   scaler = StandardScaler()
   X = scaler.fit_transform(X)
   ###
   pca_analysis = PCA(n_components=2)
   principalComponents = pca_analysis.fit_transform(X)
   ###
   pca_dict = dict()
   for index in range(len(principalComponents)):
      key =  str(int(principalComponents[index][0]*factor)) + ","
      key =  key + str(int(principalComponents[index][1]*factor))
      if key not in pca_dict:
         pca_dict[key] = list()
      if Y[index] not in pca_dict[key]:
         pca_dict[key].append(str(Y[index]))
   delete_key = list()
   for key in pca_dict:
      if  len(pca_dict[key]) < 2:
         delete_key.append(key)
   for index in delete_key:
      del pca_dict[index]
   return pca_dict;
```

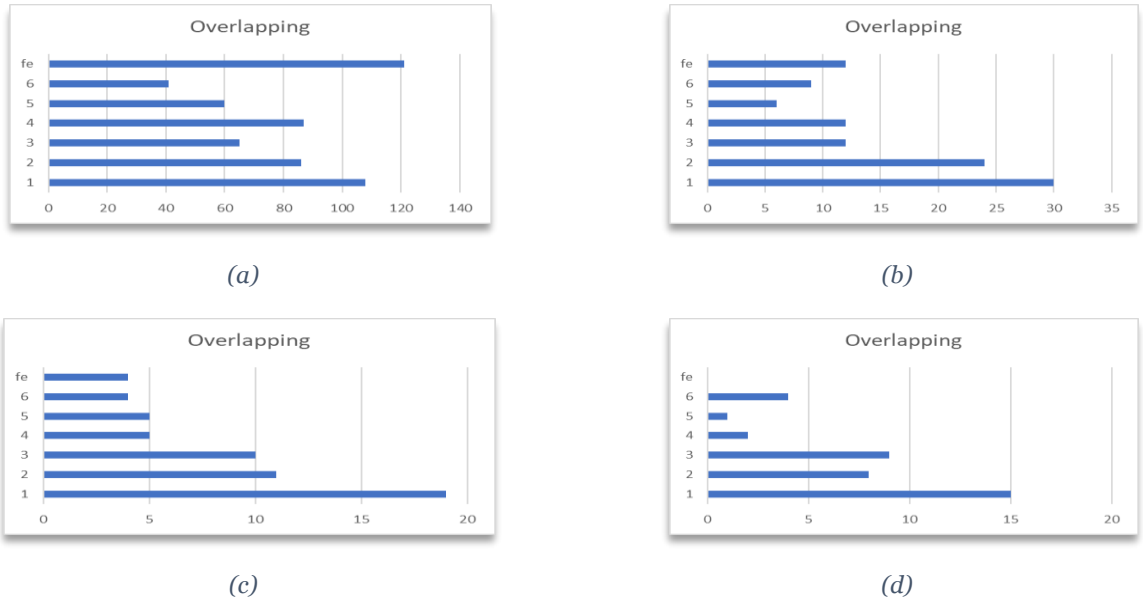
Figure 7: PCA Analysis Code

Figures 8 show the Feature extraction query winner when not found overlapping with a Factor equal to 9 x 10^3.

```
SELECT
 pk_id,dt1,dt2,dt3,dt4,
 timeMag, xMag, yMag, zMag,
 mag1,  mag2, mag3, mag4,  mag5,
 subject_tag
FROM fe_five
```

Figure 8: Feature Extraction Winner

Figures 9 show the PCA overlapping analysis. Also, FE does observe better shape when the Factor precision is incrementing. In addition, when the Factor is equal to 9 x 10^3, the FE is not overlapping. It does observe in Figure 9 (d).

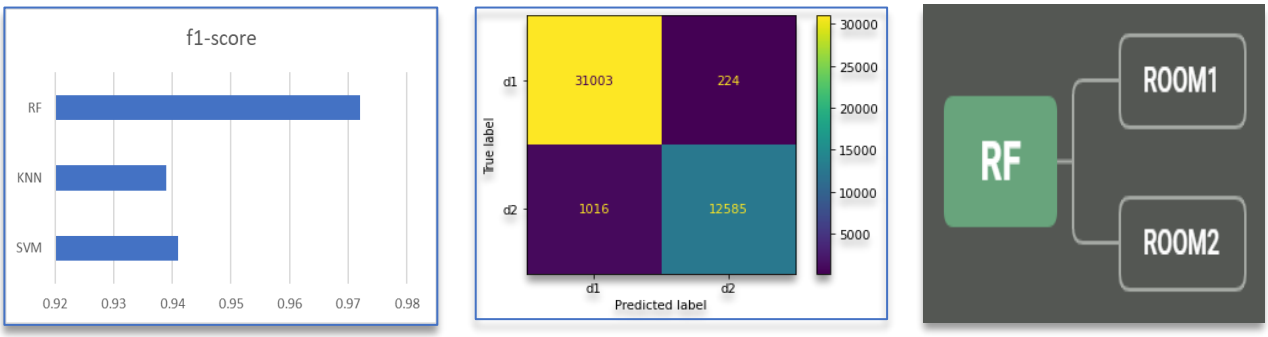


(a) *(b)*

(c) *(d)*

Figure 9: PCA Overlapping Analysis

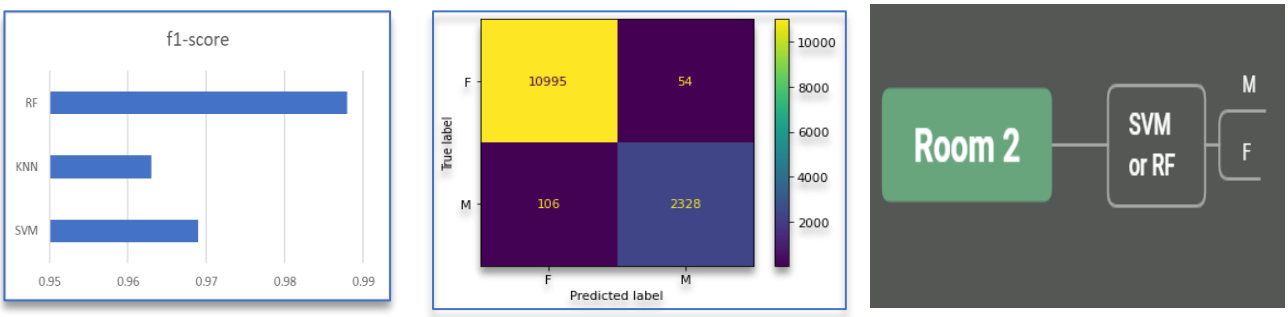
Figures 10 show the ML analysis between location classifiers between rooms 1 and 2. Also, Figure 10 (a) shows the f1-score for the ML used, and the winner is Random Forest. In addition, the ML's second position is SVM. Moreover, Figure 10 (b) is the confusion matrix from the ML winner, Random Forest. Finally, Figure 10 (c) shows the classification structure.



(a) *(b)* *(c)*

Figure 10: Machine Learning (ML) Analysis for Location Classifier

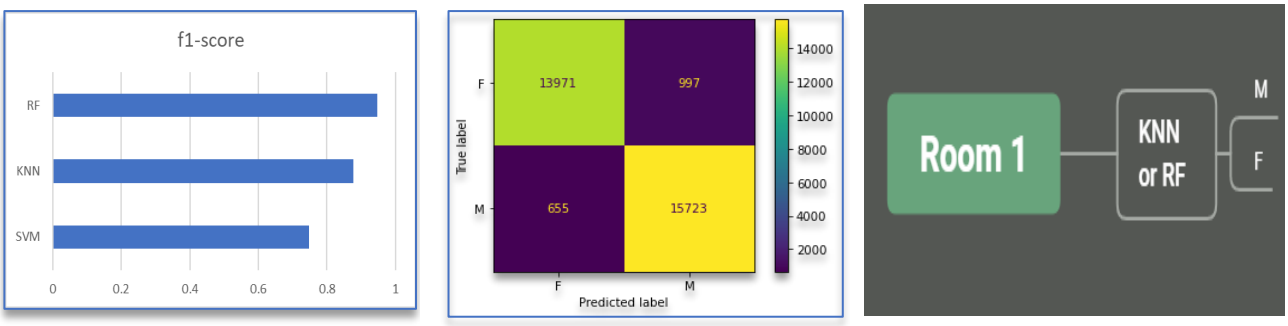
Figures 11 show the ML analysis between M & F classifiers for Room 2. Also, Figure 11 (a) shows the f1-score for the ML used, and the winner is Random Forest. In addition, the ML's second position is SVM. Moreover, Figure 11 (b) is the confusion matrix from the ML winner, Random Forest. Finally, Figure 11 (c) shows the classification structure.



(a) *(b)* *(c)*

Figure 11: Machine Learning (ML) Analysis for M & F in Room 2

Figures 12 show the ML analysis between M & F classifiers for Room 1. Also, Figure 12 (a) shows the f1-score for the ML used, and the winner is Random Forest. In addition, the ML's second position is KNN. Moreover, Figure 12 (b) is the confusion matrix from the ML winner, Random Forest. Finally, Figure 12 (c) shows the classification structure.



(a) *(b)* *(c)*

Figure 12: Machine Learning (ML) Analysis for M & F in Room 1

Figures 13 show the Hash Function Key definition for the subject tag. Also, the results are non-overlapping, which is good because it shows the importance of feature extraction, SVM, and PCA analysis.

```
CREATE TABLE last_level as
SELECT distinct
(CAST(dt1*100  AS int) || ',' ||
CAST(dt2*100  AS int) || ',' ||
CAST(dt3*100  AS int) || ',' ||
CAST(dt4*100  AS int)  || ',' ||
CAST(timeMag*100  AS int)  || ',' ||
CAST(xMag*100  AS int) || ',' ||
CAST(yMag*100  AS int) || ',' ||
CAST(zMag*100  AS int) || ',' ||
CAST(mag1*100  AS int) || ',' ||
CAST(mag2*100  AS int) || ',' ||
CAST(mag3*100  AS int) || ',' ||
CAST(mag4*100 AS int)  || ',' ||
CAST(mag5*100  AS int) ) AS key_id,
subject_tag
FROM fe_five;
```

Figure 13: Hash Function definition

Figures 14 show the machine learning structure. Also, it looks like a tree.

Figure 14: ML Structure

Conclusion

The feature extraction helps classify biometric gait authentication. Also, the combination of PCA and SVM identifies important features for constructing the hash function and overlapping pruning. The hash function helps to eliminate overlapping but needs to explore classification between Euclidean and Cosine similarity for big data. Moreover, SQLite helps a lot in managing big data.

Future Work

The research will explore the creation of KNN for big data by using the hash function definition with feature extraction and exploring Euclidean and Cosine's similarities.

Acknowledgment

The work supported is based upon this material by, or in part by, the National Centers of Academic Excellence in Cybersecurity (NCAE-C) under contract/award H98230-20-1-0411.

References

Attal, F., Mohammed, S., Dedabrishvili, M., & et al. (2015). Physical Human Activity Recognition Using Wearable Sensors. *Sensors (Switzerland)*, *15*(12), 31314–31338. https://doi.org/10.3390/s151229858

Chen, K., Zhang, D., Yao, L., Guo, B., Yu, Z., & Liu, Y. (2021). Deep learning for sensor-based human activity recognition: Overview, challenges, and opportunities. *ACM Computing Surveys*, *54*(4). https://doi.org/10.1145/3447744

Dahanayake, A., Huiskonen, J., & Kiyoki, Y. (2020). *Information Modelling and Knowledge Bases XXXI*. IOS Press. https://www.iospress.com/catalog/books/information-modelling-and-knowledge-bases-xxxi

Dong, G., & Liu, H. (2018). *Feature Engineering for Machine Learning and Data Analytics*. CRC Press. https://www.routledge.com/Feature-Engineering-for-Machine-Learning-and-Data-Analytics/Dong-Liu/p/book/9780367571856#

Raschka, S., & Mirjalili, V. (2019). *Python Machine Learning: Machine Learning and Deep Learning with Python, Scikit-Learn, and TensorFlow 2, 3rd Edition*. Packt Publishing. https://www.packtpub.com/product/python-machine-learning-third-

edition/9781789955750

Shinmoto Torres, R. L., Visvanathan, R., Hoskins, S., Van den Hengel, A., & Ranasinghe, D. C. (2016). Effectiveness of a batteryless and wireless wearable sensor system for identifying bed and chair exits in healthy older people. *Sensors (Switzerland)*, *16*(4). https://doi.org/10.3390/s16040546