SmartGlass Based Human Activity Recognition using CNN

Course code & name: PH 582: Machine Learning.
CWIDs: 12111075, 12137140.
Group: B2

Objectives:

The objective of the project is to apply the machine learning for six different human activity recognition (HAR). The previous HAR experiments showed the superiority of NN, so we used CNN and accelerometer and gyroscope sensor data to detect the following activity: walking, running, sweeping, cycling, basketball playing, and working with the computer. Using two separate devices we collected the accelerometer and gyroscope sensor data. The first device is eyeglass mounted automated ingestion monitor (AIM) device [1-2] and ActiGraph GT9X Link [3]. The performance evaluation will help to explore the smart-glass based human activity recognition.

Motivation and applications:

The purpose of human activity recognition (HAR) is to detect user behavior, such as locomotion, postures, and gestures, to understand users' habits and lifestyles and provide healthcare and wellness services for health promotion. HAR has many real-world applications, ranging from healthcare to personal fitness, gaming, tactical military applications, and indoor navigation.

Now a days, smart watches are mostly used for HAR. However, this process requires an extra equipment (watch) to wear for HAR as people hardly use watch know the time because of the availability of the smartphones. On the other hand, about 64% of adults (194.1 million) use eyeglasses for some sort of vision correction, according to The Vision Council [4]. Only 11% of them wear contact lenses and the rest uses eyeglasses. So, the AIM device is a more convenient way for HAR as it does not require any extra gadget to wear. The original AIM device was developed for the food intake behavior detection; however, human activity detection will help to monitor the energy consumption through food intake and energy deduction through activity and notify the user through smartphone notification system to achieve a balanced lifestyle.

To the best of our knowledge, this is the first time we are going to test such approach. We have also compared the performance of our proposed approach with readily available commercial product.

Previous methods:

Application of wearable sensor with CNN, LSTM for HAR is not a new concept. A lot of research have been performed over the years for HAR using machine learning algorithms. Zeng *et al*. proposed a method in which a shallow CNN is used but the HAR problem is restricted to the accelerometer data only [5]. Yang *et al*. investigate the multichannel time series data and built a new deep architecture for the CNN for HAR [6]. However, they did not create a dataset of their own and relied only on the existing dataset.

For activity recognition, Alsheikh *et al*. used triaxial accelerometers and shallow networks with handcrafted features and deep activity recognition models. But they only recognized only 6 activities [7]. Ordóñez *et al*. propose a generic deep framework for activity recognition based on convolutional and LSTM recurrent units and by assembling signal sequences of accelerometers and gyroscopes into a novel activity image, which enables Deep Convolutional Neural Networks (CNN) to automatically learn the optimal features from the activity image Jiang *et al*. proposed a method for the activity recognition task [8-9]. However, both of these methods required the user to carry an extra equipment to collect the gyroscope data for HAR.

Methodology:

In a nutshell, first we preprocessed the signal collected with AIM and ActiGraph, then used two linear classifiers to know how we should arrange the signals. Next, we created 8-bit RGB images from the selected sensor combinations and finally trained and tested CNN to classify six activities, i.e., walking, running, sweeping, cycling, basketball playing, and working with the computer. Fig. 1 delineates a high-level overview of our project.

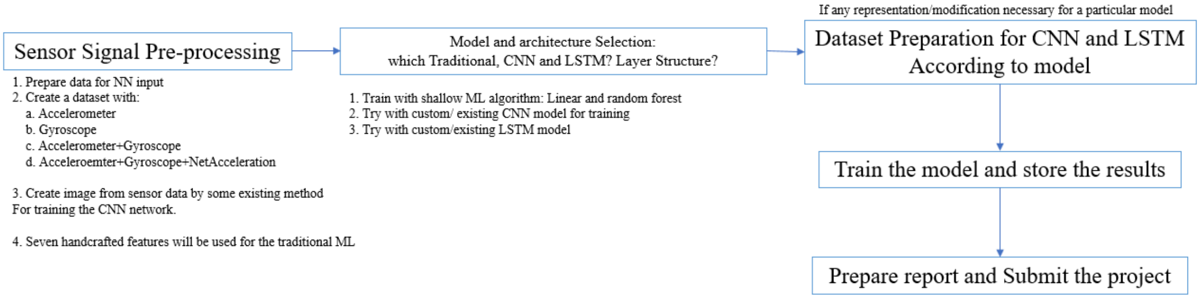
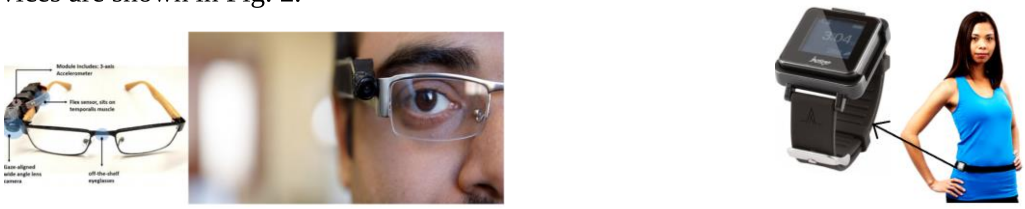

If any representation/modification necessary for a particular model
Sensor Signal Pre-processing
1. Prepare data for NN input
2. Create a dataset with:
a. Accelerometer
b. Gyroscope
c. Accelerometer+Gyroscope
d. Acceleroemter+Gyroscope+NetAcceleration
3. Create image from sensor data by some existing method For training the CNN network.
4. Seven handcrafted features will be used for the traditional ML
Model and architecture Selection: which Traditional, CNN and LSTM? Layer Structure?
1. Train with shallow ML algorithm: Linear and random forest
2. Try with custom/ existing CNN model for training
3. Try with custom/existing LSTM model
Dataset Preparation for CNN and LSTM According to model
Train the model and store the results
Prepare report and Submit the project

Fig. 1 Proposed method flowchart

Introduction to AIM and ActiGraph:

Automatic Ingestion Monitor (AIM) [12] is a wearable sensor for dietary assessment developed by Dr. Edward Sazonov, Professor, EEE, UA. It is composed of 6 degree of freedom (accelerometer, gyroscope with 128Hz sampling rate), wide angle camera and optical sensor. The device is in under research.

ActiGraph is a commercially available HAR device [13]. For this experiment, ActiGraph GT9X Link is used and it records data at 90 Hz for accelerometer and 100 Hz for gyroscope. Both of the devices are shown in Fig. 2.


Module includes: 3-axis Accelerometer
Flex sensor, sits on temporalis muscle
Gaze-aligned wide angle lens camera
off-the-shelf eyeglasses

AIM

ActiGraph

Fig. 2. AIM and ActiGraph.

Introduction to dataset:

A study has been conducted in the University of Tennessee, Knoxville, where 10 adult participant's HAR was recorded using AIM device (128 Hz accelerometer and gyroscope), attached to the right arm of a pair of eyeglasses, and using an ActiGraph GT9X (GT9X), worn on the right hip (90 Hz primary accelerometer, 100 Hz gyroscope). However, there were some annotation problems in four participant, so we did not consider them in our experiment.

The participants completed the following six structured activities for 6 minutes each 1) computer work, 2) sweeping, 3) stationary cycling, 4) treadmill walking at 3 mph (0% grade), 5) treadmill walking

6) one-vs-one basketball. The triaxial data were summarized into a vector magnitude for each sensor (VM for accelerometer and GVM for gyroscope) and collapsed to 1-second averages and the data is available for access and experimentation.

Data Preprocessing and Image conversion

We started with up-sampling ActiGraph's accelerometer data from 90Hz to 100Hz for uniformity. Apart from using only signals from accelerometer, we also calculated the net acceleration using the eq. 1 for our project to see whether it improves the performance or not.

$$\begin{aligned} \text{nacc\_}x &= \sqrt{y^2 + z^2} \\ \text{nacc\_}y &= \sqrt{x^2 + z^2} \\ \text{nacc\_}z &= \sqrt{x^2 + y^2} \end{aligned} \tag{1}$$

Here, *x*, *y*, *z* corresponds to the three channels of gyroscope and accelerometer signals.

In the next step, we calculated the vector magnitude of the of the Accelerometer and Gyroscope data using 10 seconds epoch. We also calculated seven basic features (mean, median, minimum, maximum, STD, variance and mean to STD ratio) from the vector magnitude. Then we used the following classifiers to get the primary results.

1. Linear Classifier implemented by MATLAB fitcdiscr (describe those MATLAB function)

2. Random Forest implemented by MATLAB fitcensemble

Table. I. Primary results:

Classifiers	**Sensor**	**AG Dataset; Epoch=10s**	**AIM Dataset; Epoch=10s**
		Accuracy	Accuracy
Linear Classifier	Acc	0.8705	0.7964
	Gyr	0.5806	0.5593
	Acc+Gyr	0.8770	0.8334
Random Forest	Acc	0.8970	0.7916
	Gyr	0.6052	0.6006
	Acc+Gyr	0.9005	0.8556

Here, Acc is accelerometer data, Gyr is gyroscope data, and Netacc is net-acceleration.

As primary results showed combining sensor data improved performance, for CNN, we created three types of images using the following sensor data combination:

1. Accelerometer (acc_x, acc_y, acc_z) + Gyroscope (gyr_x, gyr_y, gyr_z)
2. Net Acceleration (nacc_x, nacc_y, nacc_z) + Gyroscope (gyr_x,gyr_y,gyr_z)
3. Accelerometer (acc_x, acc_y, acc_z) + Gyroscope (gyr_x, gyr_y, gyr_z) + Net Acceleration (nacc_x, nacc_y, nacc_z)

Before image conversation we have limited our gyroscope data between -500 to 500 as it has some high outlier values (showen in Fig. 2 for basketball activity) which will affect the image conversion.

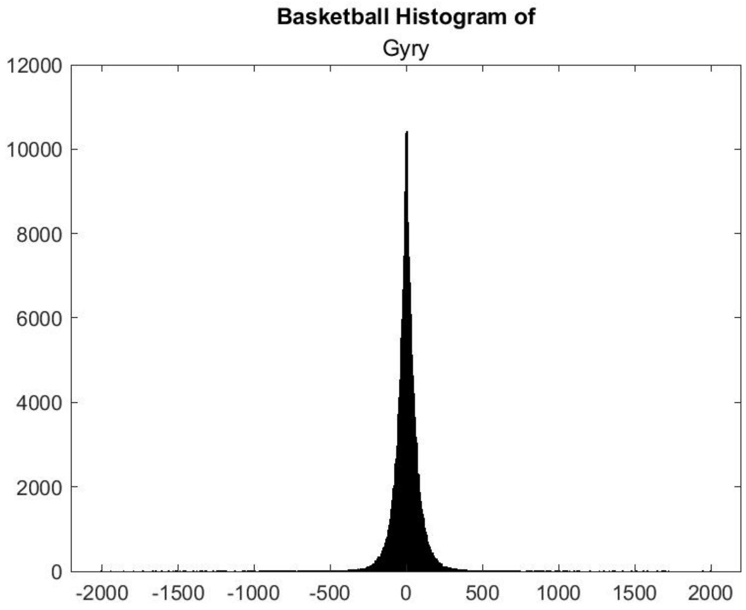


Fig. 2. Histogram of Gyroscope data of Basketball activity for ActiGraph.

Then we normalize all signals and scaled them to 255 using eq. 2 to create 8-bit image.

$$\begin{gathered}\overline{x} = \frac{x-min(X)}{max(X)-min(X)} \times 255 \\ \overline{y} = \frac{y-min(Y)}{max(Y)-min(Y)} \times 255 \\ \overline{z} = \frac{z-min(Z)}{max(Z)-min(Z)} \times 255\end{gathered} \tag{2}$$

Here, x, y, z corresponds to the three channels of gyroscope and accelerometer signals.

To cope with the decimal values, using the encoding technique given in eq. 3, we converted the normalized signals into three integers that corresponded to the pixel values in red, green, and blue channels of a color image.

$$\begin{gathered}R_{\overline{x}} = \lfloor \overline{x} \rfloor \\ G_{\overline{x}} = \lfloor (\overline{x} - \lfloor \overline{x} \rfloor) \times 10^2 \rfloor \\ B_{\overline{x}} = \lfloor \left(\overline{x} \times 10^2 - \lfloor \overline{x} \times 10^2 \rfloor\right) \times 10^2 \rfloor\end{gathered} \tag{3}$$

Here, $\bar{x}, \bar{y}, and\ \bar{z}$ are the normalized sensor values.

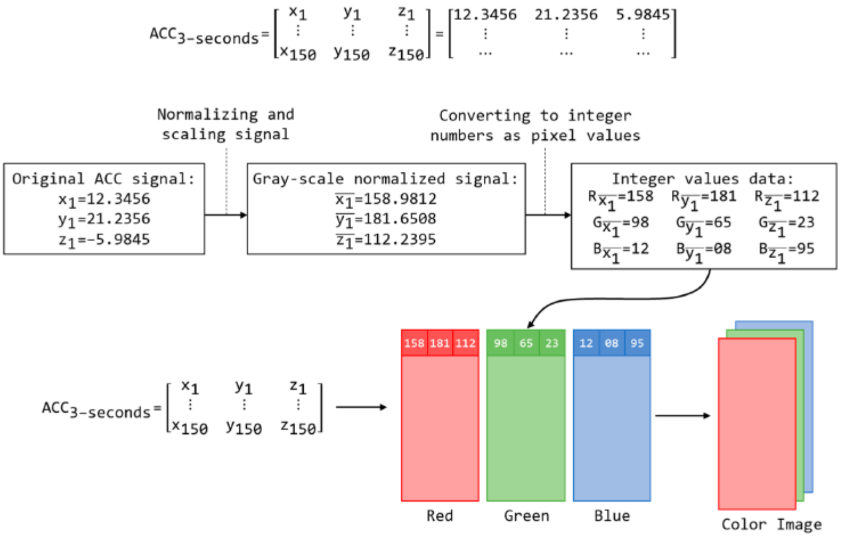

ACC3-seconds = [x1 y1 z1; ⋮ ⋮ ⋮; x150 y150 z150] = [12.3456 21.2356 5.9845; ⋮ ⋮ ⋮; … … …]
Normalizing and scaling signal
Converting to integer numbers as pixel values
Original ACC signal: x1=12.3456 y1=21.2356 z1=-5.9845
Gray-scale normalized signal: x1=158.9812 y1=181.6508 z1=112.2395
Integer values data: Rx1=158 Ry1=181 Rz1=112 Gx1=98 Gy1=65 Gz1=23 Bx1=12 By1=08 Bz1=95
158 181 112
98 65 23
12 08 95
Red
Green
Blue
Color Image

Fig. 3. RGB image creation process illustration [10].

Fig. 3 illustrates the RGB image creation process. The images that were created using first two combination had 6 columns as there were 9 channels and the imaged that were created last combination created had 9 columns for similar reason.

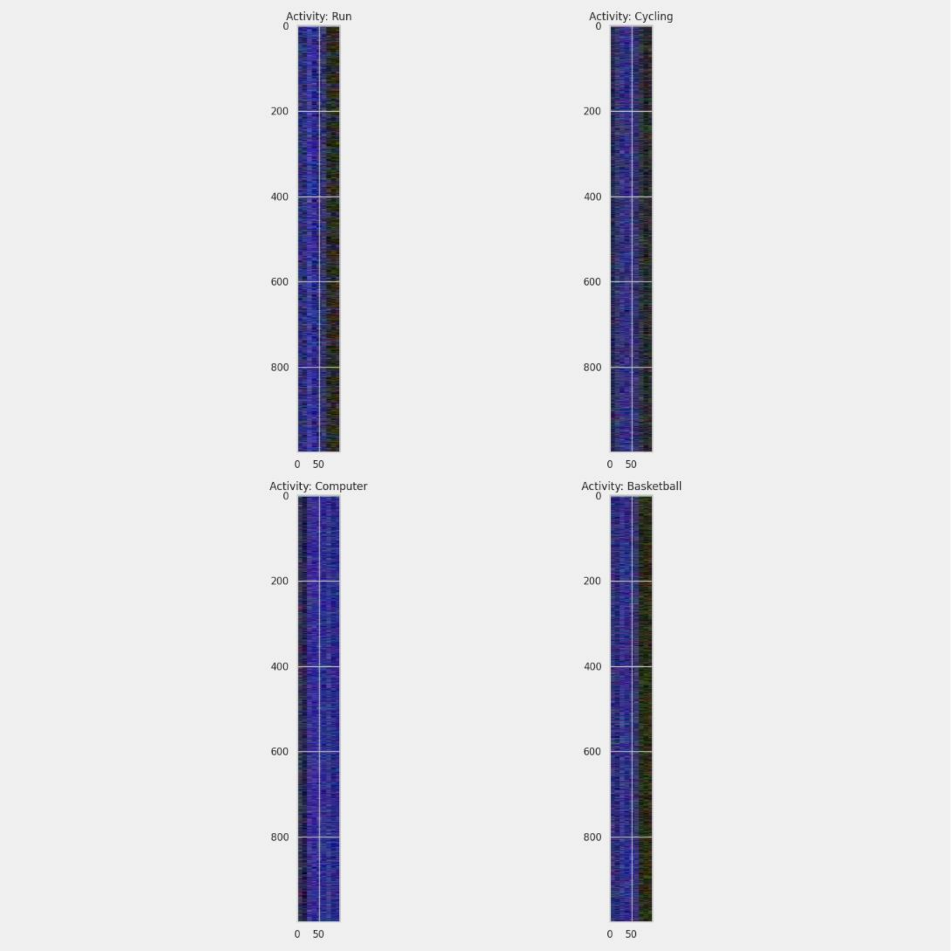


Fig. 4. Sample image of 10s epoch, Acc+Gyr+Noracc sensor combination (ActiGraph)

After that, for each activity and for each sensor combination, we extracted 500 random chunks of 2s, 5s and 10s data from and converted into images. For instance, for 10s epoch, the third sensor combination of ActiGrapg device data image had 9 columns and (10 X 100 Hz) or 1000 rows and thus creating an image with a dimension of 9 X 1000, and 500 unique images were created for this category for one activity and (500 X 6 activities) or 3000 images were created for each subject. We collected data from six subjects and so a total of (6 X 3000) or 18,000 images were needed for each category. Thus, for two devices, we had to create a total of 162,000 images. A sample image is given in Fig. 4. Creating image in this way was inspired by a study of Hur *et al* [10].

CNN Model:

There were four 2-D convolutional layer and one fully connected dense layer in our CNN architecture. The four 2-D convolutional layers had the following square filter sizes: 2,3,2,3 respectively. Each layer stride size of 1, and padding size of 1. After each convolutional layer, there were a batch normalization layer to normalize each input channel across a mini-batch, a dropout layer for regularization, and a ReLU layer as the activation function. The dropout started at 10% and was increased by 10% after each layer. There were four blocks of convolutional operation, and each block included two modules of {2Dconvolutional and batch-normalization, dropout, and ReLU}. The modules differ in the number of filters defined inside a convolutional layer: the first module had 64 filters, the second module had 128 filters, the third module had 256 filters, and the second module had 512 filters. Between the convolutional blocks there was a average pooling layer to perform down-sampling by dividing the input into rectangular pooling regions. The first and third average pooling layers in the network had a square pooling region size of 2, stride size of 1, and padding size of 1, the second and third average pooling layers in the network had a square pooling region size of 3 and same stride and padding as the previous pooling layers.

The fully connected layer had four layers before softmax function. The layers had 512, 1024, 1024, and 128 neurons respectively and ReLu as activation function. Again, Batch normalization and dropout was added after each layer in the previous manner. Fig. 5 illustrates our CNN model. [colab link]

Dataset split and parameters for training:

We had six subject's data whom we denoted as 101, 102, 103, 104, 105, and 106. The dataset was splitted in the following method:

Repeat for All Subject:

- Use one subject as testing.
- Rest of the subject image divide with 75% train and 25% testing.
- Save the results.

For example: 2s spoch of any sensor combination for Actigraph had a total of 18000 images. The, subject 101's 3000 images was used for testing and the rest of the 15000 images from other subjects were split in 75% (11250) for training and 25% (3750) for validation.

The number of maximum epochs were 80 and we employed early stop with patience 20. The training was performed with batch size of 500, and the initial learning rate was 0.00002 which was halved after 10 epochs. For optimizer, we used SGD optimizer with lr=0.001 and momentum 0.9. Categorical cross entropy was used as loss function. As balanced dataset we focused on accuracy and loss metric and used leave one out cross validation. Please follow this colab link to use the model [11].

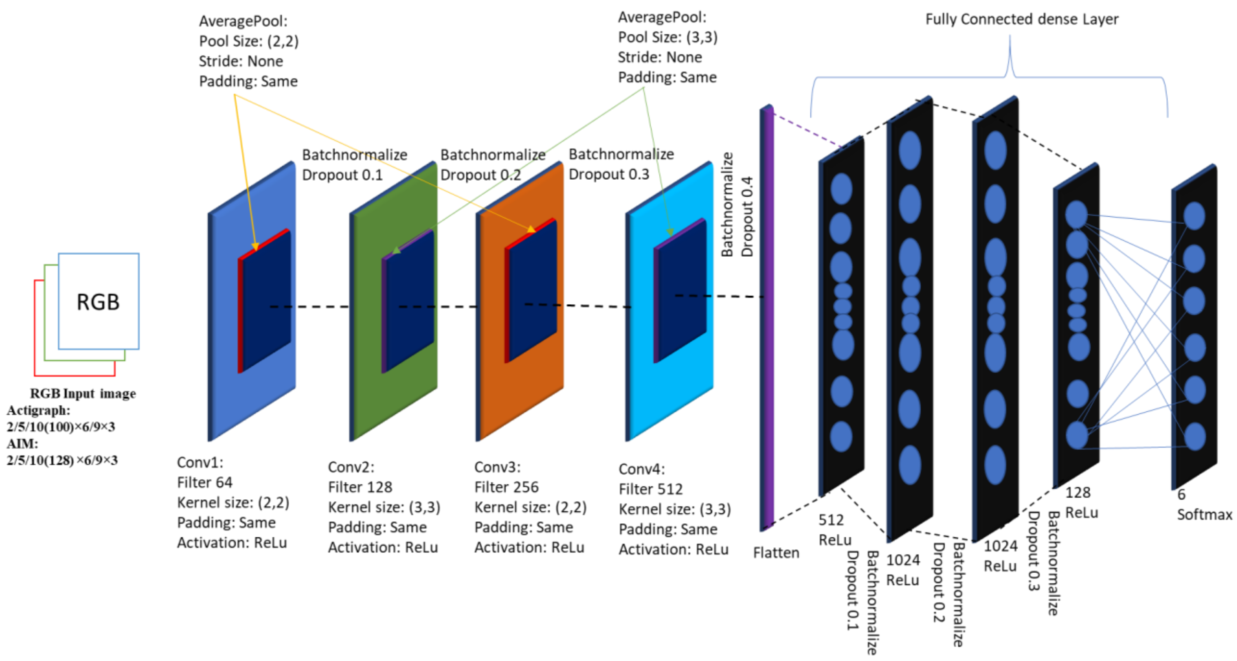


Fig. 5. CNN model

Result:

The following tables shows a performance comparison of the devices:

Table II. Performance of AIM device

Metric (Average)	***AIM***								
	Acc+Gyr			***Netacc+Gyr***			***Acc+Netacc+Gyr***		
	2s	***5s***	***10s***	***2s***	***5s***	***10s***	***2s***	***5s***	***10s***
Training Accuracy	0.9546	0.9823	0.9948	0.9808	0.9954	0.9991	0.9954	0.9996	0.9995
Training Loss	0.1491	0.0607	0.0247	0.0729	0.0275	0.0136	0.0296	0.0123	0.0115
Validation Accuracy	0.9509	0.9840	0.9928	0.9793	0.9938	0.9990	0.9945	0.9994	0.9999
Validation Loss	0.1596	0.0564	0.0299	0.0773	0.0306	0.0141	0.0291	0.0123	0.0121
Testing Accuracy	0.9315	0.9676	0.9820	0.9708	0.9885	0.9929	0.9901	0.9968	0.9976
Testing loss	0.2259	0.1048	0.0608	0.1084	0.0515	0.0331	0.04238	0.0232	0.0229

Here, Acc is accelerometer data, Gyr is gyroscope data, and Netacc is net-acceleration.

Table III. Performance of ActiGraph device

Metric (Average)	***ActiGraph***								
	Acc+Gyr			***Netacc+Gyr***			***Acc+Netacc+Gyr***		
	2s	***5s***	***10s***	***2s***	***5s***	***10s***	***2s***	***5s***	***10s***
Training Accuracy	0.9595	0.9737	0.9745	0.8480	0.9283	0.8859	0.9040	0.9278	0.9115
Training Loss	0.1089	0.0714	0.0733	0.3673	0.1684	0.2730	0.2125	0.1709	0.2098
Validation Accuracy	0.9586	0.9726	0.9709	0.8528	0.9287	0.8867	0.9036	0.9272	0.9108
Validation Loss	0.1132	0.0731	0.0786	0.3565	0.1743	0.2738	0.2079	0.1772	0.2098
Testing Accuracy	0.8147	0.8531	0.9709	0.8385	0.9192	0.8610	0.8967	0.9053	0.9036
Testing loss	0.8472	0.9503	0.0786	0.4299	0.1734	0.3656	0.2287	0.2080	0.2028

Here, Acc is accelerometer data, Gyr is gyroscope data, and Netacc is net-acceleration.

From these tables we can find the best sensor combination and sliding window time in the following manner.

Increasing the sliding window time increases the number of datapoint and accuracy (Fig. 6). However, 2s is sufficient to achieve HAR 96% accuracy with AIM.

The average best accuracy (99.48% for AIM and 90.18% for ActiGraph) was found while we combined all type of data i.e., accelerometer, gyroscope, and net-acceleration. Fig. 8 shows the confusion matrix and accuracy per epoch for subject no. 106 as test subject with sliding window of 10s and for combining all three types of data collected from AIM device.

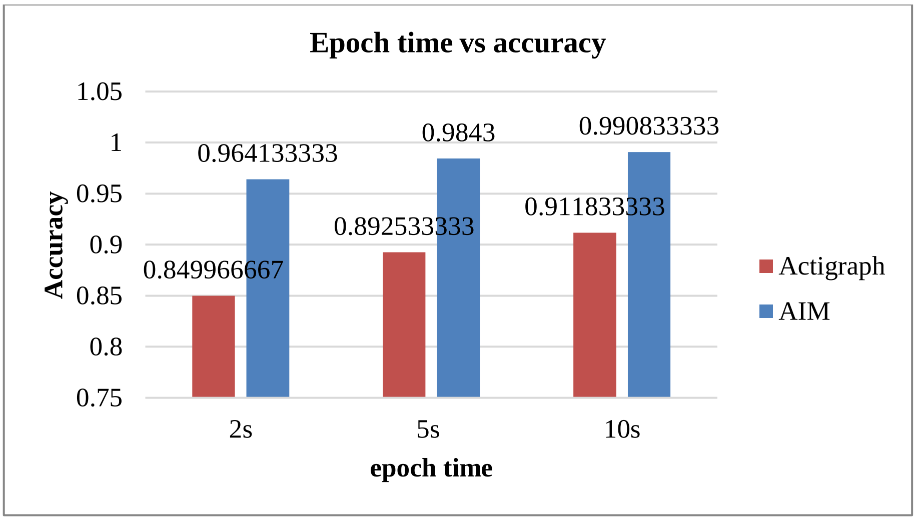


Fig. 6. Epoch time vs accuracy

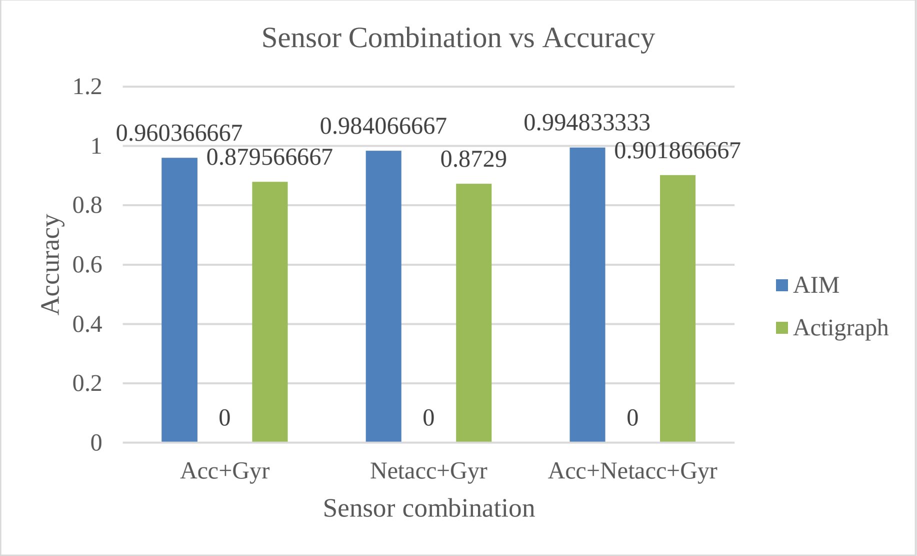


Fig. 7. Sensor combination vs accuracy

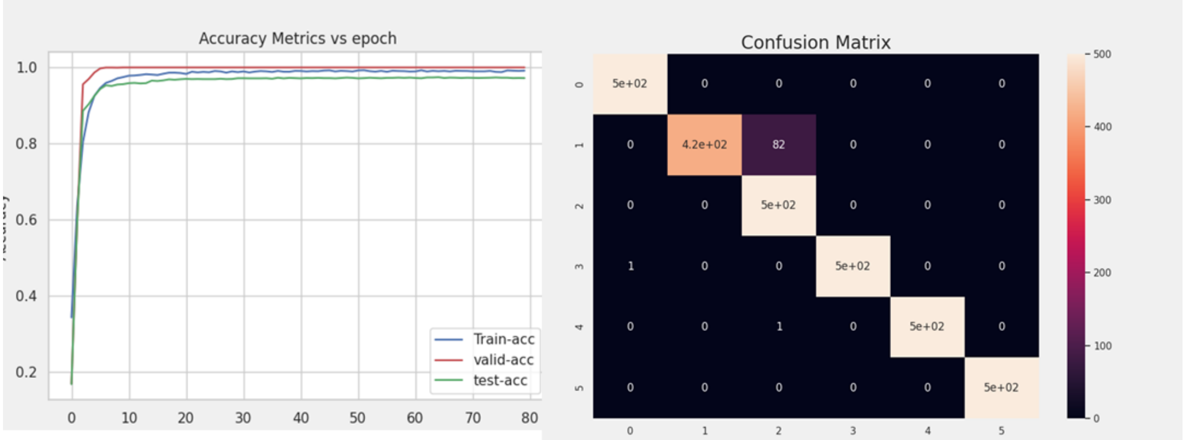


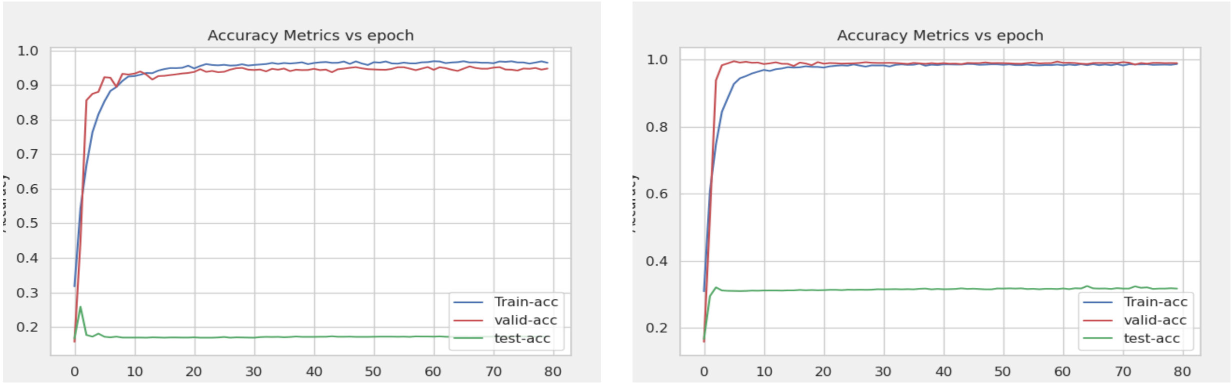
Fig. 8. Accuracy vs epoch and confusion matrix. Labels ['Basketball' 'Computer' 'Cycling' 'Run' 'Sweeping' 'Walk'] is represented [0 1 2 3 4 5]

However, the model confused (major misclassification) Run with walking. Because the original study has running with different speed, and we merge that into one class. Some running is slow which confused with walking activity.

Also, some unexpected issues happening during training and testing for ActiGraph 101 test subject with 2s and 5s acc+gyr combination (Fig. 9). However, the accuracy of 10 second is good enough.

Conclusion:

CNN has high accuracy for HAR detection. Which will lead a new platform for the smart glass based HAR recognition. The more the epoch (data), the better the performance. However, it is possible to detect the HAR with 2s data with 96% accuracy. The analysis shows that the Acc+Gyr+Netacc data combination is superior to other combinations. Finally, the AIM performs better than ActiGraph. Although some issues have to be delt with, yet, AIM opens a new platform for HAR.



2s sliding window

5s sliding window

Fig. 9. Accuracy Vs epoch for 2s and 5s sliding window.

In future, we are planning to plot the PCA and other feature extraction method to see the features of anomalous subject's data to solve previously mentioned issues. Also, we will use LSTM and CNN+LSTM combination for improving performance. Finally, we plan to validate our model with real time data which will provide more confidence about out result.

References:

[1] Fontana, Juan M., Muhammad Farooq, and Edward Sazonov. "Automatic ingestion monitor: a novel wearable device for monitoring of ingestive behavior." *IEEE Transactions on Biomedical Engineering* 61.6 (2014): 1772-1779.

[2] Doulah, A. B. M. S. U., et al. "“Automatic Ingestion Monitor Version 2”—A Novel Wearable Device for Automatic Food Intake Detection and Passive Capture of Food Images." *IEEE Journal of Biomedical and Health Informatics* (2020).

[3] https://actigraphcorp.com/actigraph-link/

[4] https://www.allaboutvision.com/eyeglasses/faq/why-people-wear-glasses/#:~:text=About%2075%25%20of%20adults%20use,according%20to%20The%20Vision%20Council.

[5] Ming Zeng, Le T. Nguyen, Bo Yu, Ole J. Mengshoel, Jiang Zhu, Pang Wu, and Joy Zhang. Convolutional neural networks for human activity recognition using mobile sensors. In MobiCASE, 2014.

[6] Yang, Jianbo, Minh Nhut Nguyen, Phyo Phyo San, Xiaoli Li, and Shonali Krishnaswamy. "Deep convolutional neural networks on multichannel time series for human activity recognition." In Ijcai, vol. 15, pp. 3995-4001. 2015.

[7] Alsheikh, Mohammad Abu, Ahmed Selim, Dusit Niyato, Linda Doyle, Shaowei Lin, and Hwee-Pink Tan. "Deep activity recognition models with triaxial accelerometers." arXiv preprint arXiv:1511.04664 (2015).

[8] Ordóñez, Francisco Javier, and Daniel Roggen. "Deep convolutional and lstm recurrent neural networks for multimodal wearable activity recognition." Sensors 16, no. 1 (2016): 115.

[9] Jiang, Wenchao, and Zhaozheng Yin. "Human activity recognition using wearable sensors by deep convolutional neural networks." In Proceedings of the 23rd ACM international conference on Multimedia, pp. 1307-1310. 2015.

[10] Hur, Taeho, Jaehun Bang, Jongwon Lee, Jee-In Kim, and Sungyoung Lee. "Iss2Image: A novel signal-encoding technique for CNN-based human activity recognition." Sensors 18, no. 11 (2018): 3910.

[11] Project colab link: https://colab.research.google.com/drive/1JnMbD_s--LIQCkkwGPS9p6oB2FPtQC5j?usp=sharing