CS205 HEALTH ANALYTICS

HUMAN ACTIVITY RECOGNITION

Team members:
Karan Sanwal (205028682)
Shweta Sood (905029230)
Sonali Garg (104944076)

INTRODUCTION

Today, most smartphones and smartwatches are equipped with motion and direction sensors, which has led to Human Activity Recognition (HAR) based on wearable sensors (accelerometer, gyroscope, magnetometer, etc.) becoming widely important due to its ease and large number of promising applications. Ubiquitous identification of physical activity, i.e. recognising and monitoring activities can help assist patients with chronic diseases like obesity. According to research, monitoring activity patterns could help prevention of many health disorders in children. Another interesting example is Home Care Monitoring, which allows disabled and elderly patients a continuous health and well-being supervision while they perform Activities of Daily Living (ADL) at home.

In this project, we aim to collect and use sensor data from a smartwatch to recognise the activity a person is currently performing. We specify 4 activities that we aim to recognise i.e. sitting, standing, walking and laying down, and mainly focus on data from triaxial accelerometer to do so. We try various preprocessing techniques, features and machine learning models in order to achieve our purpose.

LITERATURE REVIEW

In this section, we review some of the research in the area of HAR. Kwapisz et al.[1] worked on Activity Recognition using Cell Phone Accelerometers. They collected labeled accelerometer data from twenty-nine users as they performed daily activities such as walking, jogging, climbing stairs, sitting, and standing, and then aggregated this time series data into examples that summarize the user activity over 10- second intervals. They generated various features like Average acceleration (for each axis), Standard Deviation (for each axis), Average Absolute Difference(for each axis), Average Resultant Acceleration(norm), Time Between Peaks, and Binned Distribution, and used logistic regression, decision trees (J48)and multilayer neural networks for classification purposes. The perceptron seemed to work best for them, however they used cross-validation, hence, not really cross-person.

White et al.[3] worked on estimation of physical activity energy expenditure during free-Living from wrist accelerometry in UK adults. They used a combined sensor and a wrist accelerometer.(60Hz triaxial) The combined sensor measured heart rate and trunk acceleration, which was combined with a treadmill test to yield a signal of individually-calibrated PAEE. They used Vector magnitude(VM), or Euclidean Norm, as a feature, which can be interpreted as the magnitude of acceleration the device was subjected to at each measurement, including gravitational acceleration. They also realised that there will also be a potential sensor noise component in the high frequency domain (above human physiological range), which they filtered out by a 20 Hertz low-pass filter. They calculated two derivatives of VM, both aiming to remove the gravity component from the signal in order to isolate the activity-related acceleration component; 1) Euclidean Norm Minus One (ENMO) subtracts 1g from VM and truncates the result to zero at sample level, whereas 2) High-Pass Filtered Vector Magnitude (HPFVM) applies a high-pass filter to the VM signal at 0.2 Hertz, therefore treating gravity as a low-frequency component to be filtered out. These two signals, ENMO and HPFVM, are both plausible approximations of acceleration as a result of human movement, and are the primary descriptions of wrist acceleration used in the paper.

Liu et al.[4] worked on computational methods for estimating energy expenditure in human physical activities. They considered time as well as frequency domain features(as well as other features like age), and models like SVM for their task. We got the intuition for using frequency domain features through this paper.

Mannini et al.[5] worked on activity recognition in youth using single accelerometer placed at wrist or ankle. They again used the norm to get signal independent of the orientation of the sensing node. This was low pass filtered using a 15 Hz cut-off 4th order Butterworth filter to limit the bandwidth of the signal to the frequencies common in human motion To classify data within the four defined activity classes, they divided the data into 12.8 s non-overlapping windows. Their work however focussed on using data from adults and enhancing it through new features to make it work for youth.

Bersch, Sebastian D., et al. [6] did a survey on Sensor data acquisition and processing parameters for human activity classification. This paper was highly useful in that it spoke of a variety if preprocessing techniques and how they compared to each other, mainly for AAL(Ambient Assisted Living). This gave a great starting point in what all could be done and what was more likely to be useful.

IMPLEMENTATION DETAILS

Dataset Description

Smartwatch data collection involved periodic activity logging for the activities: sleeping, walking, standing, sitting at a sampling frequency of 250 Hz.

1. Training data: 7.6 Million
    a. Sitting samples: 2.6 Million
    b. Walking samples: 1.98 Million
    c. Standing samples: 1.58 Million
    d. Laying Down: 1.38 Million
2. Testing data 1: 4.2 Million samples
3. Testing data 2: 2.2 Million samples

For the purpose of Human Activity Recognition, we utilized the smartwatch's triaxial accelerometer data as this was noted to be the most crucial for the task.

Accelerometer Data Scatter

It made intuitive sense to first see the scatter of data to investigate the type of classifier to be used. Based on the scatter, it seems that no complexity classifiers would be able to get a good accuracy on this data as a lot of overlapping exists. Even non-parametric methods like K-Nearest Neighbours would perform poorly. Hence, feature generation needs to be performed, to achieve a classifiable scatter.

Pre-processing

Pre-processing involved stripping of the first and the last 10 seconds of data. This was done because right at the beginning and towards the end of the activity there will be some sudden jerks which can corrupt the normal logging of activity. Thus, it becomes important to remove this noise.

Following the stripping of data, we performed two major types of data aggregation:

1. Sub-sampling
    a. We randomly chose 20 samples for each second of data for each sensor. We did so because the literature on this topic agreed that 250 samples per second were not needed and just increased computational load. While many researchers have taken a sampling frequency of about 50, it was seen that minimal gains were seen in the classification task on increasing sampling frequency over 20. We corrected the skew in our data by under-sampling the classes that much more samples than the others in our training data.
2. Windowing
    a. For every posture, we extracted contiguous five second windows for feature extraction. This was also done for ten second windows, but we got best results with five second windows

Feature Selection

For a window of five seconds the following features were constructed:

1. Mean: Commonly used feature, including the work done by Kwapisz et al[1]. Used for smoothing the data
2. Median: Since mean is affected by outliers, we wanted to try median
3. Standard Deviation: Commonly used feature, including the work done by Kwapisz et al [1]
4. Norm: $\sqrt{(x^2+y^2+z^2)}$ Commonly used feature under different names, including the work done by White, Tom, et al [2]
5. Pitch: $\mathrm{atan}(y/\sqrt{(x^2+z^2)})$ Handles intra-class variability according to Agrawal et al [3]
6. Roll: $\mathrm{atan}((-x)/z)$ Handles intra-class variability [3]
7. Range
8. Energy of the highest and second highest frequency after Fast Fourier Transform - Liu, Shaopeng et al [4]

Model Selection

For model selection, inspired by the Occam's Razor, we thought of starting with the naive distance based models.

Distance Based Models

Curse of High Dimensionality states that as the feature space increases, distance becomes less and less intuitive for taking decisions. Infact, distance as a notion starts breaking down in higher dimensions. To verify this notion, we checked if distance as a metric suffices along the axes with maximum variance in the data. This may give us insight into what may happen in the higher dimensions.

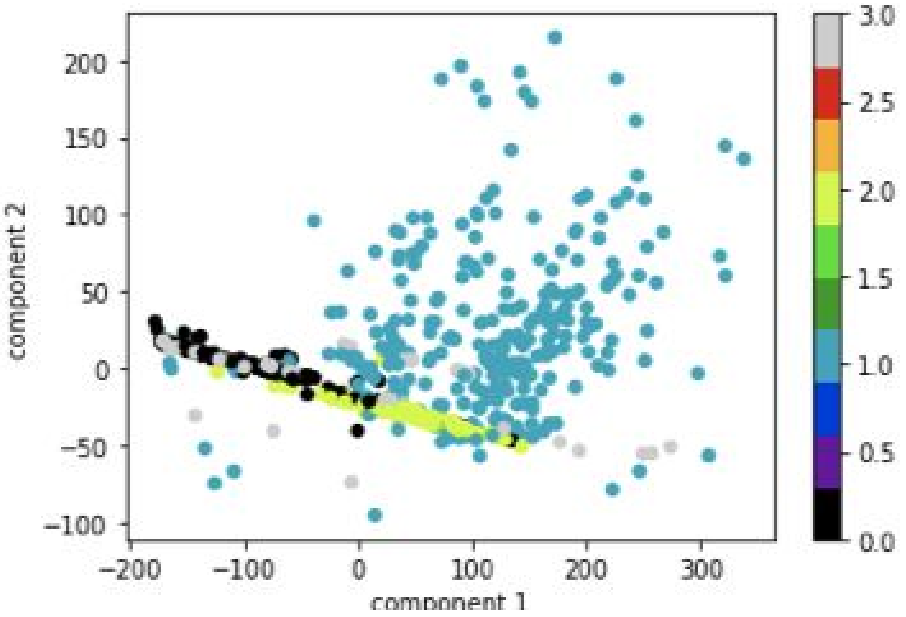


Fig. 1 : Plot of accelerometer data along the axes with maximum variance

As can be seen from the above figure, the distance based metric, even aided with boosters like Dynamic Time Warping (DTW - for measuring similarity between two temporal sequences) is heavily skewed. Thus, this measure is expected expected to perform poorly.

To further test our intuition, we randomly picked 2000 training and testing examples (both balanced) and ran a simulation of K Nearest Neighbor with DTW 5 times. The reason for taking a small sample size was that KNN with DTW is an expensive algorithm complexity-wise for large volume of data. So it made sense to first test it on a smaller sample size.

As expected, we got a low accuracy since only a few training examples were used, but the model should still be able to fit into some trends, which were absent.

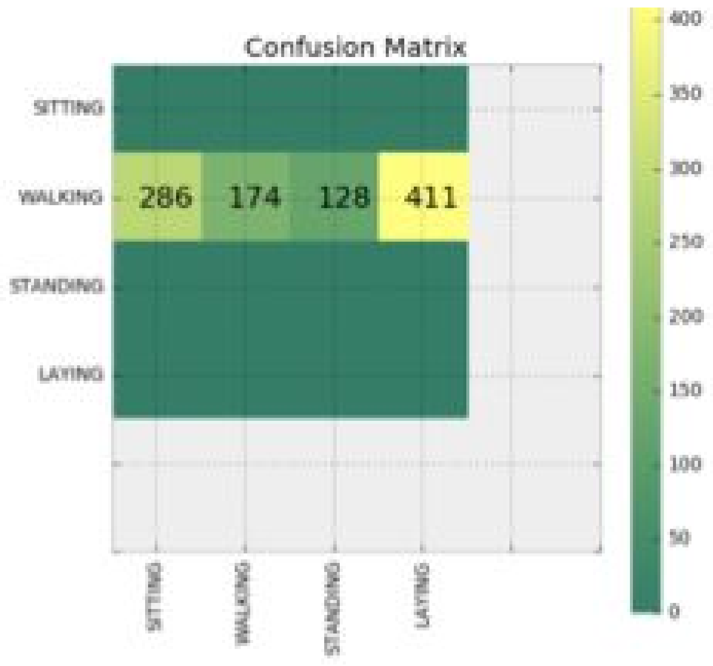


Fig. 2 : Confusion Matrix of testing samples classified all as walking with KNN + DTW

All the testing samples were classified as walking. This resonates with what we saw in Fig. 1 Since the projected data is highly overlapped, it has a high likelihood of being classified to a common class.

We then compared the performance of this model with a weak neural network of only 1 layer with 10 hidden node trained on the same data.

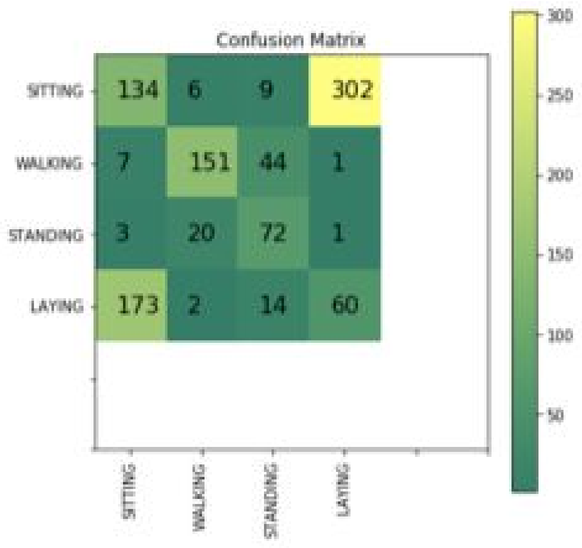


Fig. 3 : Confusion Matrix of 1 layer neural network

The simple neural network still performed better than the distance based model. Hence, we ruled out the distance based models from further consideration.

Next, we evaluated Gradient Based Model.

Gradient Based Models

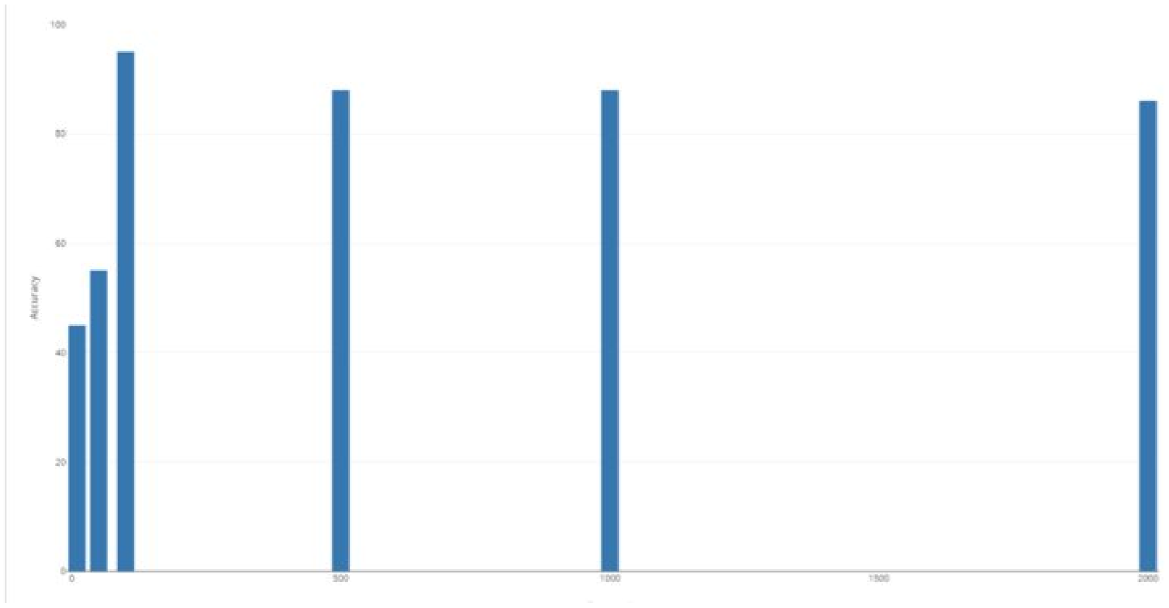
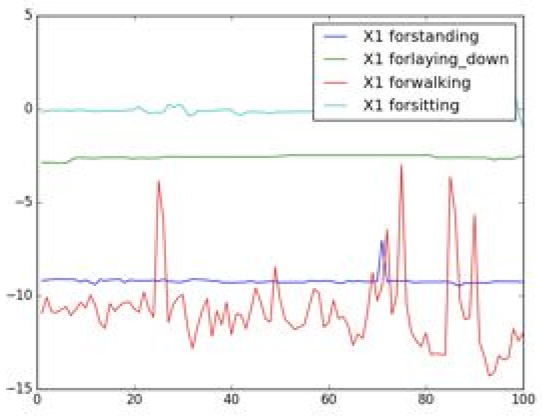
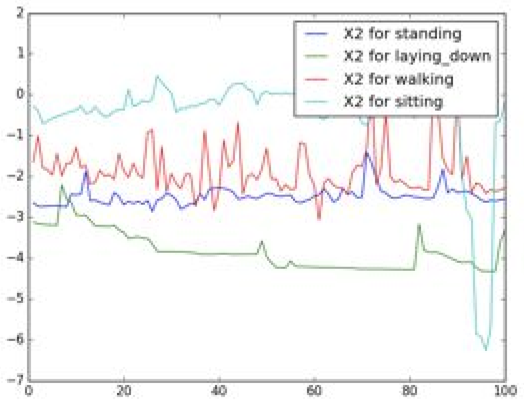


Fig.4 : Training accuracy vs Number of nodes in Network

The following figure shows how a Neural Network might be a good classifier to classify on this data. An accuracy of 86% is reached when the test set belongs to the same person whose data it is trained on. But we experimented with using neural networks on cross-person train test [training on one person and testing on others], and the accuracy dropped drastically. Even when the number of nodes were increased, the test accuracy saturates. Accuracy Saturates at around 46% when tested on data collected from a different person. This makes sense because, a neural net is a powerful tool for extracting pattern and in this case, the pattern is for a specific person. As most researches point out, to train a generalized neural net, a varied training set is an absolute compulsion.

Entropy Based Models





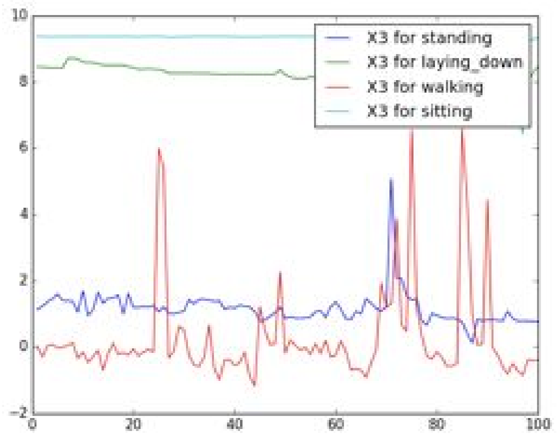


Fig. 5 : Temporal plot of every axis of accelerometer

To understand the dynamics of the data provided to us, we plotted accelerometer data vs time along all the 3 axis of accelerometer data [X1, X2, X3]. This helped in developing our next intuition. As we can see from the graphs above, X1, X3 is crucial for distinguishing between standing, walking versus laying down, sitting. It might hence be intuitive to use entropy based classifiers as they are used for building a tree-like structure, dividing the sample space into 'purer' subsamples based on an entropy rule.

To start with, we tested Random Decision Forests. It is a widely used machine learning technique well known for its efficiency over large datasets. Random Forest is an ensemble of Decision Trees used for classification of data by bootstrap aggregating and feature bagging. To classify a new vector, the input is passed down each of the trees in the ensemble and they vote for a class. Final prediction is made based on the majority vote in the ensemble.

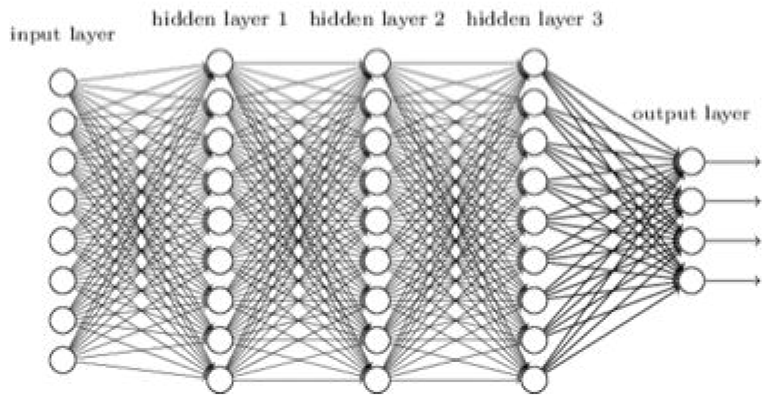
For the purpose of classification, the following combination of features performed the best: mean, standard-deviation, energy of of highest and second highest frequency after FFT.
The classifier gave an accuracy as high as 96.4% for the same person; whereas for cross-person it was as high as 74.5%. More details are described in the Results section.

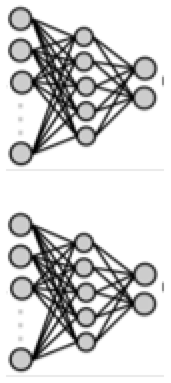
Since entropy based models performed so well, we decided to explore more ensemble based classifiers:

- Calibration based Training : Self Proposed Time optimized Deep Learning Model

- Generalized Training : Using Gradient boosted Trees
- 2-Level Ensembling : Self Proposed Conditional Ensembling

Calibration based Training : Self Proposed Time optimized Deep Learning Model





Conventional Deep Learning Architecture

Proposed Classifier Architecture

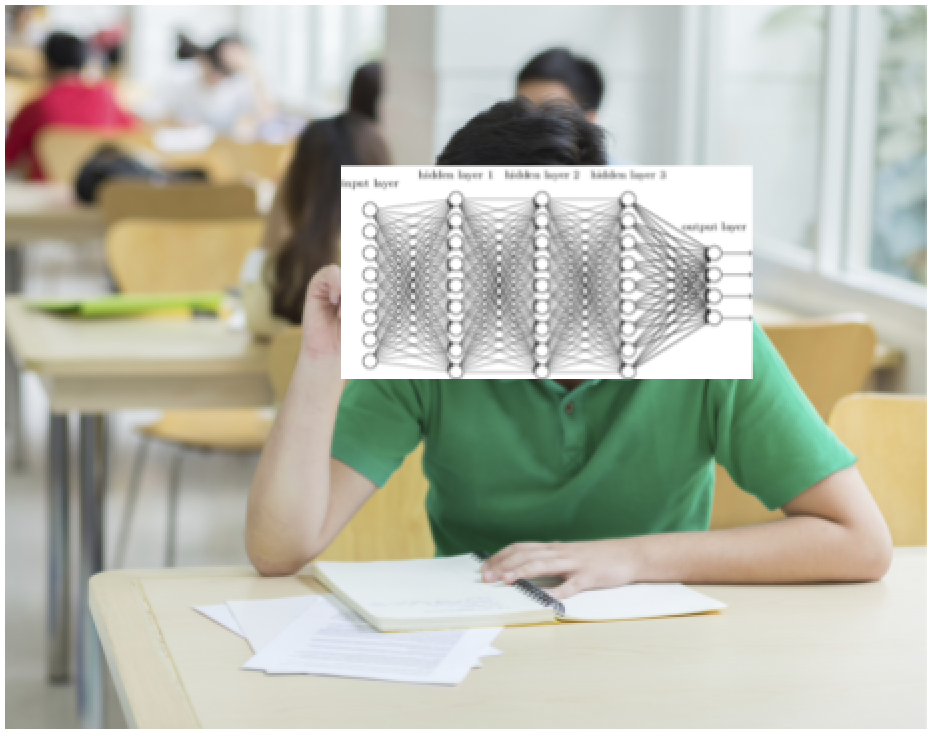


Fig. 6 : Traditional CNN

As we can from Fig. 6, a traditional CNN does not realize that it can communicate with other Neural Networks to solve the exam, so has to work harder to score.

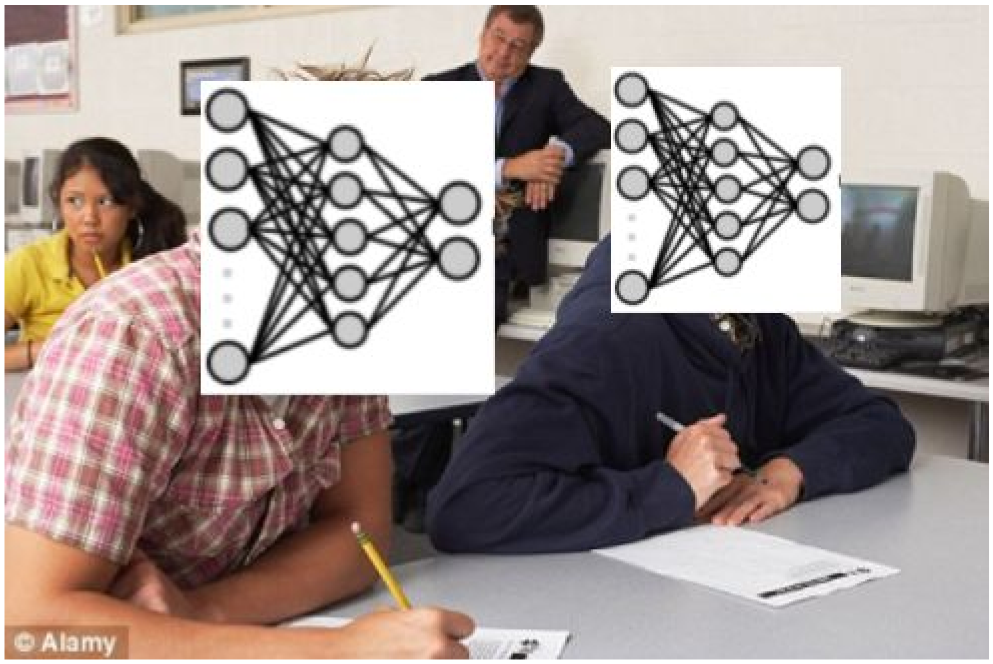


Fig. 7: Proposed CNN

Whereas, Fig. 7 leverages the fact that cheating and communication is allowed during the exam, so don't have to work that hard. It is inspired from the techniques of student collaboration in a group project. Just like a single student working on a project requires more time and effort and rather a collaboration would save time, effort in achieving appreciable accuracy over hard tasks.

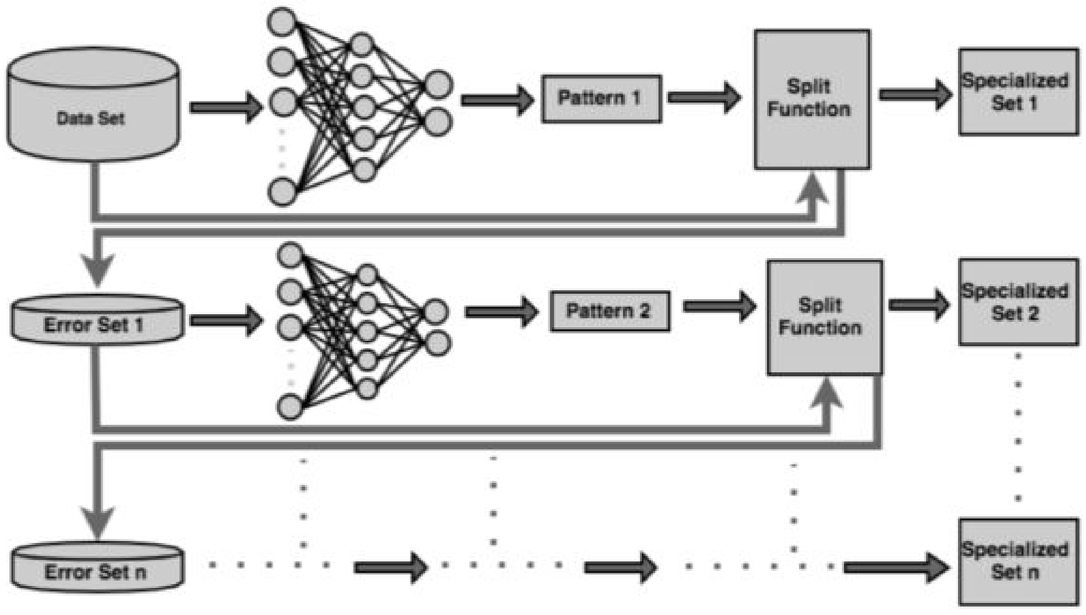


Fig. 8 : Architecture of the Proposed Model

As seen from the figure above, the error set of one network becomes the training set of others, so in a way, we are cresting specialized networks each specializing in one particular set of data. At test time, we just have to develop a 'teacher-function' to direct the problem to the right neural network. This is done via analyzing the confidence of the networks, and arranging the networks in order of their generalization and going down the order unless we find a really confident neural network for that particular problem.

As investigated earlier, deep learning is only good for personalized training/testing on the same person. This could hence be an efficient and computationally cheap method to train a strong calibration based classifier. We effectively achieved an accuracy of 90% on the same person train-test data.

Generalized Training : Using Gradient boosted Trees

The promising results using decision tree led us to try XGBoost.

XGboost is a machine learning technique used in a variety of applications to generate predictive models from large quantities of data. It considers an ensemble of various prediction models that are weak and enhances them. Generally, decision trees are the weak learners that are used in the process of gradient boosting. The trees are generated in an additive model where a gradient descent algorithm is used to minimize a differentiable loss function.

Using the same features as RDF, we got same person accuracy as high as 95.2% and cross person upto 85.74%. More details are described in the Results section.

XGBoost Parameters Used:

The overall parameters have been divided into 3 categories by XGBoost:

- **General Parameters:** Guide the overall functioning
  - booster: gbtree (tree based model as opposed to linear)
  - nthread: defaults to max number of threads available(for parallelization)
- **Booster Parameters:** Guide the individual booster (tree/regression) at each step
  - learning_rate: 0.1
  - min_child_weight: 1 (used to control overfitting)
  - max_depth: 3 (The maximum depth of a tree, used to control overfitting))
  - Gamma: 0 (minimum loss reduction required to make a split)
  - lambda : 1(L2 regularization term on weights)
- **Learning Task Parameters**: Guide the optimization performed
  - objective: multi:softprob (multiclass classification using the softmax objective, returns predicted probabilities)
  - eval_metric :merror – Multiclass classification error rate

2-Level Ensembling : Self Proposed Conditional Ensembling

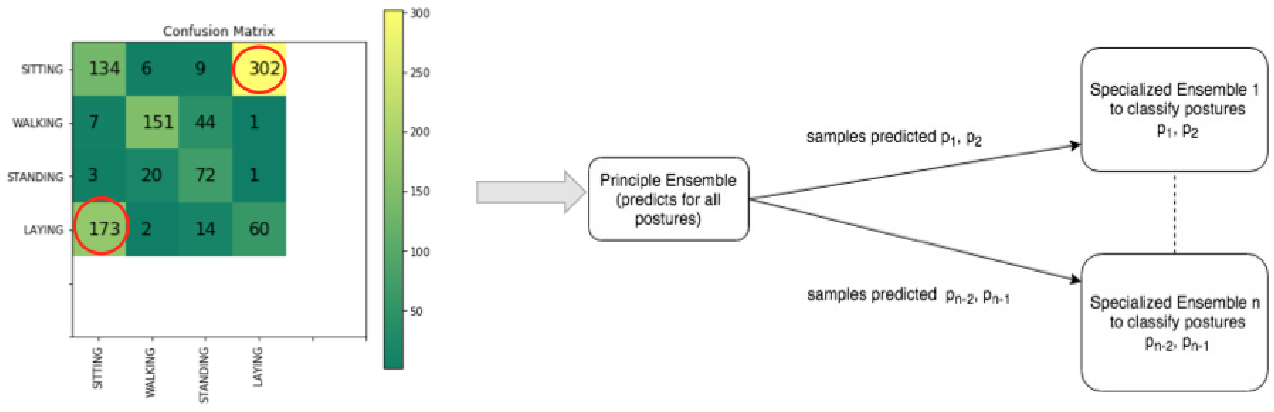

Confusion Matrix
SITTING
WALKING
STANDING
LAYING
134
6
9
302
7
151
44
1
3
20
72
1
173
2
14
60
300
250
200
150
100
50
Principle Ensemble (predicts for all postures)
samples predicted p_1, p_2
samples predicted p_{n-2}, p_{n-1}
Specialized Ensemble 1 to classify postures p_1, p_2
Specialized Ensemble n to classify postures p_{n-2}, p_{n-1}

Fig. 9 : Conditional Ensembling

Conditional Ensembling is a technique to ensemble classifiers in terms of their specialization all connected to a generalized centric model that directs the problem statement to the right classifier. This is a method that almost guarantees the accuracy to shoot up. The intuition is that the bigger problem is broken into smaller problems with each classifier specializing in that particular subdomain, (so in this case one classifier for deciding between standing-walking and the other to classify between sitting-lying down). Then the generalized classifier is asked to solve the problem, and let's say if it predicts 'sitting', the the classifier specializing between sitting-lying down is asked to make the final prediction. This helps remove the errors that were induced by a sitting being predicted as lying down or vice versa. Fig. 9 shows this example in terms of confidence matrix.

RESULTS

RDF

Number of estimators = 100

For dataset 1:

Mean accuracy score: 0.62
Confusion Matrix:

	Sitting	Walking	Standing	Laying down
Sitting	1052	36	26	47
Walking	21	605	52	0
Standing	54	156	301	8
Laying down	957	79	31	425

For dataset 2:

Mean accuracy score: 0.75
Confusion Matrix:

	Sitting	Walking	Standing	Laying down
Sitting	322	3	0	10
Walking	104	80	7	35
Standing	39	40	32	45
Laying down	61	101	63	1050

XGBoost

For dataset 1:

Mean accuracy score: 0.86
Confusion Matrix:

	Sitting	Walking	Standing	Laying down
Sitting	1060	19	2	80
Walking	19	585	73	2
Standing	54	139	315	11
Laying down	125	25	0	1342

For dataset 2:

Mean accuracy score: 0.74
Confusion Matrix:

	Sitting	Walking	Standing	Laying down
Sitting	313	8	0	14
Walking	104	86	2	34
Standing	32	43	26	55
Laying down	95	76	61	1043

Conditional Ensembling

For dataset 2:

Mean accuracy score: 0.75
Confusion Matrix:

	Sitting	Walking	Standing	Laying down
Sitting	317	8	0	10
Walking	104	86	2	34
Standing	32	43	26	55
Laying down	76	76	61	1062

DISCUSSION

Development of a Human Activity Recognition (HAR) system is composed of two main modules: a feature extractor for obtaining the most relevant characteristics from the inertial signals every second, and a machine learning algorithm for classifying between the different activities.

For feature extraction, the following combination of features performed the best: mean, standard-deviation, energy of of highest and second highest frequency after FFT.

For classification, Gradient Boosted Trees gave the best performance at 86%, outperforming Neural Network based architectures. This shows that neural networks aren't necessarily intelligent feature extractors for every problem. Even random decision forests couldn't match upto XGBoost and stood at 75%. The major reason for the high accuracy of XGBoost is in terms of the training objective. Boosted Trees try to add new trees that compliments the already built ones. This normally gives better accuracy with less trees. This is based on the idea of reinforced learning ,i.e., learning from the mistakes of previously trained classifier rather than hustling to the next one without any learning. The time series plots of accelerometer data corroborate that entropy based ensemble classifiers will perform better on this data.

Moreover, we proposed two architectures: Time optimized Deep Learning Model and two level conditional ensembling. While the former is great for calibration based training method, the other is a performance booster. The former actually implements computationally cheap deep learning architectures. While the latter goes for smarter ensembling techniques. Both these tend to show a lot of potential in terms of ensembling techniques for posture classification.

Future work will focus on developing new stream-based algorithms that can update the model quickly and on the fly. It is also desirable to have light algorithms that enable us to update the model on the smartwatch while reducing the amount of power consumption. Furthermore, noise and outlier detection algorithms for streams of data will be improved and applied to our algorithm.

TSFRESH

- A python based library to extract features for time-series data for classification/regression tasks
- Extracts as much as 250 features from the data and then performs feature selection to only filter out relevant features
- A comprehensive list of feature extractors and selectors is made available but run time complexity is too high to be run on a single node, which is why we could not leverage the use, but this is something we would definitely like to try given a cluster of nodes

REFERENCES

[1] Kwapisz, Jennifer R., Gary M. Weiss, and Samuel A. Moore. "Activity recognition using cell phone accelerometers." *ACM SigKDD Explorations Newsletter* 12.2 (2011): 74-82

[2] White, Tom, et al. "Estimation of physical activity energy expenditure during free-living from wrist accelerometry in UK adults." *PLoS One* 11.12 (2016): e0167472

[3] https://www.andrew.cmu.edu/user/kdagrawa/documents/har.pdf

[4] Liu, Shaopeng, Robert Gao, and Patty Freedson. "Computational methods for estimating energy expenditure in human physical activities." *Medicine and science in sports and exercise* 44.11 (2012): 2138.

[5] Mannini, Andrea, et al. "Activity Recognition in Youth Using Single Accelerometer Placed at Wrist or Ankle." Medicine and science in sports and exercise 49.4 (2017): 801-812.

[6] Bersch, Sebastian D., et al. "Sensor data acquisition and processing parameters for human activity classification." *Sensors* 14.3 (2014): 4239-4270