Recognizing and Classfying Human Activity Using Sensors

1st Ryan Barnes-Batterbee
Faculty of Engineering, Environment and Computing
Coventry University
Coventry, England
batterbr@uni.coventry.ac.uk

***Abstract*—This paper provides and discusses the results of applying different Machine Learning techniques to a data set, to classify human activities that are performed by eight human subjects wearing inertial and magnetic sensors. In this paper three classification techniques are implemented and compared: a decision tree, a k-nearest neighbors algorithm and a support vector machine algorithm. The performance of the classification techniques is compared by providing and discussing their expected classification accuracy on testing data, which is visualized using confusion matrices. Their performance is validated using 10-fold cross-validation. The dataset was originally created by recording the activities of 8 human subjects who were wearing 9 sensor units on their chest, each arm and each leg. The dataset contains 9120 5-second samples of recorded activity, for the 8 subjects performing 19 different daily and sport activities. Each instance has 5625 features: 125 instances of recorded data for the 45 sensors in each 5 second segment. Principal Component Analysis was used to reduce the large dimensionality of the data, by extracting 50 features (the first 50 principle components) which explained over 70% of the variance in the data.**

I. Introduction - Recognizing Human Activity

Human activities are naturally ambiguous making correctly recognizing a human performing any given activity a challenging problem for machines. Any given activity can be performed in different styles by different people, and people can perform multiple activities simultaneously. Hence, there is a huge amount of information not being extracted from society due to the limited ability for machines to recognize human behaviour. If this difficulty was overcome and a machine could quickly analyse and recognize human activity, there would be various significant applications. For instance, surveillance systems could detect unusual and suspicious behaviour, and the robotics industry could create machines that better respond to human activity [1].

The decreasing production costs of Microelectromechanical systems (MEMS), due to technological advancements, has made MEMS sensors more affordable for consumer purposes. This has created new realistic applications for the use of inertial sensors, which previously were not possible due to the high cost of the sensors [2]. Such applications include the potential to recognize and monitor human activity of people wearing sensors on their body [3]. This application is increasingly popular in health care as it can be used to remotely monitor the physically/mentally disabled, children and the elderly. It also allows diagnostics from health care services to be given closer to patients, such as at their home or local clinic, since the sensors are portable [4,5]. Despite the increased affordability and target market size for MEMS sensors, the research space for activity recognition is still largely focused on vision based systems with video footage [6]. However, the practical applications of visual activity recognition are limited outside of a confined space such as the house of a willing subject. The video capture of a person may be agreed to legally by a willing subject, but it invades the privacy of other humans the subject interacts with, which can have legal repercussions [7,8]. MEMS sensors however can be subtly placed on a humans body without interfering with the privacy of non-willing participants.

Most studies attempting to recognize human activity using non-visual means focus on classifying distinct activities such as: walking, jogging and standing [9]. However, Barshan has contributed research papers [10,11,12] which discuss the successfulness of various Machine Learning techniques when it comes to recognizing 19 different human activities. The human activity was recorded using three types of equipment: Inertial sensors which provide dynamic motion information, Gyroscopes which provide angular rate information around an axis, and Accelerometers which provide linear/angular velocity information. The data set the author discusses, classifies human activities that can be as subtly different as lying on your back or lying on your right side, and as widely different as sitting and playing basketball. The author had great success classifying the data using various techniques, after they reduced its dimensionality using the statistical manipulation of data.

This paper discusses and compares the results of applying three Machine Learning techniques to the dense data Barshan discussed and provided. This data was chosen since the classification of human activity using motion sensors is relatively unexplored compared to classification using visual methods. The large amount of data collected allows the effectiveness of Machine Learning techniques to be fully demonstrated, whilst also highlighting potential problems that can occur when applying them. In section II the data set is discussed in more detail. In section III the Machine Learning classification techniques which were applied to the data set are briefly discussed. In section IV the steps required to pre-process the data are discussed. In section V and VI the results of applying the Machine Learning techniques to the data set are presented and discussed. The appendix includes Jupyter Notebooks which can be followed and re-run to accurately reproduce the results.

II. The Data Set

The data set contains data collected using miniature inertial and magnetic sensors. 4 male and 4 female willingly participating subjects between the age of 20 and 30, performed 19 different activities. The subjects were not strictly instructed how to perform the activities, and instead were asked to perform activities in their own style. The total signal duration is 5 minutes for each activity of each subject. The sensor units were calibrated to retrieve data at a 25 Hz sampling frequency. The 5-minute signals are divided into 60 5-second segments. Therefore, for each activity 480 signal segments were obtained.

9 sensors were assigned to the subjects torso, right arm, left arm, right leg and left leg. Each set of 9 sensors retrieved x, y, z coordinates for accelerometers, gyroscopes, and magnetometers for that limb. When retrieving data at a 25 Hz sampling frequency, data is retrieved 125 times in each 5 second signal segment. Therefore, each signal segment has $45 \times 125 = 5625$ features. This paper explores how to best apply machine learning techniques to these signal segments to classify them as one of the 19 activities.

Table I shows the list of activities subjects performed. Note the significant similarity between certain events regarding semantics. Both activitys A3 and A4 required participants to lie down, since participants were not given strict instructions how to perform each action, we would expect that these two actions to be difficult to differentiate without visual observation. We would also expect activities A2, A7 and A8 to be difficult to differentiate without visualization since the only difference is the location that a subject is standing in. Similarly, we would expect activities A9, A10 and A11 to be difficult to differentiate since these activities require a subject to be walking in different environments. In addition to this we would expect similarity in the sensor data between A5 and A6, since these two activities involve walking up/down stairs, as well as similarity between A15 and A16 since these two activities require cycling on an exercise bike. We also note how significantly varied some activities can be when performed over a 5-minute duration. For instance, we would expect A19 to be hard to classify due to the various required acts a subject could perform when playing basketball.

It is due to this combination of subtle difference between activities, and the loose instructions on how to perform them, that this data set provides a practical challenge for Machine Learning. In the real-world activities are not performed by every person the same exact way, and the difference between activities performed by a human may have very little physical difference. For instance, standing (A2) and standing in an elevator (A7/A8) are semantically different and easy for humans to differentiate. However, an individuals may behave only slightly different in these three situations, which would be hard for visual methods to notice. When applying Machine Learning techniques to this dataset we wish to correctly classify similar activities, to demonstrate the effectiveness of using Machine Learning to recognize human activities using non-visual methods.

TABLE I
List of Activities Performed

Activity ID	***Activity Description***
A1	Sitting
A2	Standing
A3	Lying on back
A4	Lying on right side
A5	Ascending stairs
A6	Descending stairs
A7	Standing in an elevator
A8	Standing in an elevator (moving)
A9	Walking in a parking lot
A10	Walking on a treadmill with a speed of 4 km/h
A11	A10 in a 15-degree inclined position
A12	Running on a treadmill with a speed of 8 km/h
A13	Exercising on a stepper
A14	Exercising on a cross trainer
A15	Cycling on an exercise bike (horizontal position)
A16	Cycling on an exercise bike (vertical position)
A17	Rowing
A18	Jumping
A19	Playing basketball

III. Classification Techniques

A. *Decision Tree*

A decision tree algorithm classifies instances by following a sequential procedure. When classifying an instance, a decision tree algorithm traverses a binary tree from the root node to a leaf. The leaf that the traversal ends at determines the classification of the instance. The route from root to leaf is decided in segments, at each node the algorithm arrives at, the algorithm must decide if the instance meets a certain condition or not. For instance, a node rule could be 'If the value for attribute 1 is less than 0.5, visit the left child. Else, visit the right child.'. To classify the given data set, a binary decision tree is generated using an optimised version of the CART algorithm [13]. A decision rule for each node is decided by finding the rule that minimizes the impurity (Gini impurity) of the node.

B. *K-Nearest Neighbours (KNN)*

The KKN algorithm is a lazy learner that simply memorizes the training data and learns how to appropriately classify each test instance when it is presented [14]. A KKN algorithm classifies instances by locating the k-nearest instances in the training data. The most common value for the target value amongst those k instances is then chosen as the classification for any new instance. Therefore, KNN classification is very sensitive to the local structure of data. Hence, the chosen quantity of nearest neighbours located is important for the algorithm to correctly classify test instances. However, there is no mathematical formula to determine the correct value of k and therefore when using a KNN algorithm to classify the given data set, k Is chosen by experiment with different values and trying to maximize the correct classification rate over multiple values. Note that this algorithm uses the Euclidean distance to describe the distance between each instance of data, and therefore it is important that during data pre-processing, the data is scaled appropriately (normalized/standardized).

C. *Support Vector Machine (SVMs)*

If the features of data are not linearly separable, an SVM algorithm can pre-process the data and represent the features in a higher-dimensional space in which they can become linearly separable [15]. The data is mapped onto this higher-dimensional space using a kernel function, where data can then be classified with linear decision surfaces. Note that since this algorithm also makes use of distance calculations involving the instances of data, it is important to scale data appropriately during pre-processing for SVMs models. To classify the given data set, a linear and gaussian kernel are used, with the linear kernel providing much better results for this data set.

IV. Experimental setup

The original data was submitted to the UCI Machine Learning Repository inside a folder called Data. This folder has 19 sub folders, which themselves have 8 sub folders. The 19 subfolders represent the 19 classes of activity. The final 8 sub folders contained 60 text files, each containing 125 lines of comma separated values for the 45 feature columns. This data was combined into a single text file with 9120 lines, where each line represents the 5625 ($45\ attribute\ columns \times 125\ samples$) features of an instance, and the class of an instance. Note that the data is perfectly class balanced with 480 recorded instances of each activity, and that there was no missing data in this data set.
The features where then scaled using standardization. Therefore, each attribute was transformed into a new attribute with mean 0 and unit variance. The data required feature scaling since the raw data of the 45 feature columns varies widely in term of size. Therefore, when applying certain Machine Learning techniques to this data such as those that calculate the distance between two points, the techniques will not work correctly since the features will not be weighted equally.
The original data had 5625 features, hence the data needed dimensionality reduction to decrease the computational complexity of building models using it. Therefore, once the data was standardized, Principle Component Analysis (PCA) [16] was used for dimensionality reduction. Figure 1 shows the first 350 principal components. Figure 2 and 3 show the cumulative explained variance of the data sat for the first 350 and 50 principal components retrospectively. Whilst the first 350 principle components explain approximately 90% of the variance, the first 50 components explain approximately 70%. Due to the reduction of 5575 dimensions, extracting only the first 50 components when they explain such a large amount of the variance is justified. Therefore, the first 50 components are used as the features conducting classification experiments.

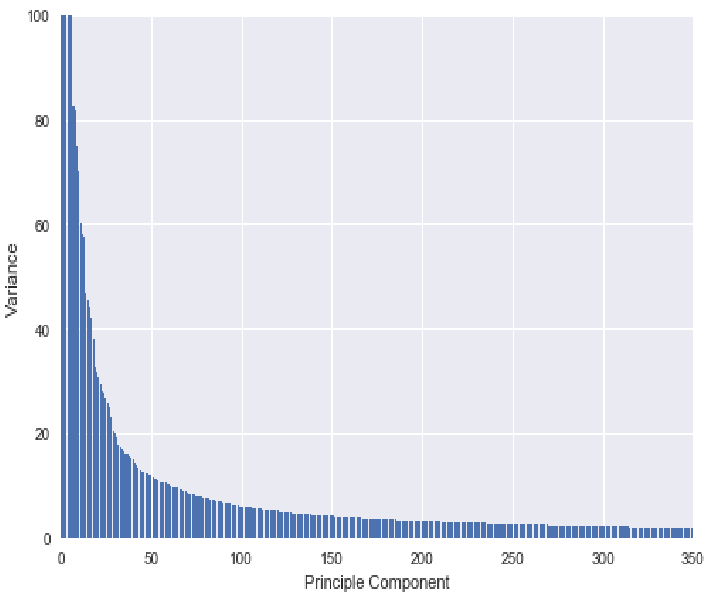

Fig. 1. The first 350 principal components.

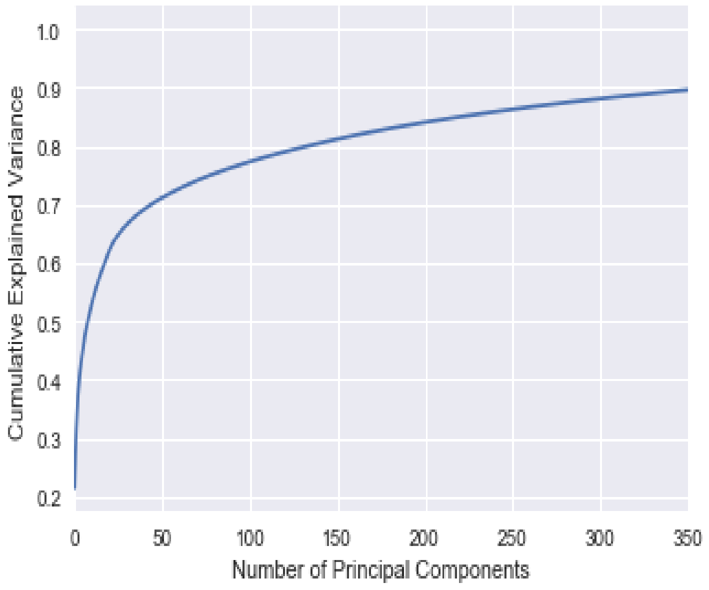

Fig. 2. The cumulative explained variance for the first 350 principal components.

V. Experiment Results

The classification techniques described previously are used to classify the 19 different activities. A total of 9120 instances with 50 features (selected by PCA) are available to use as

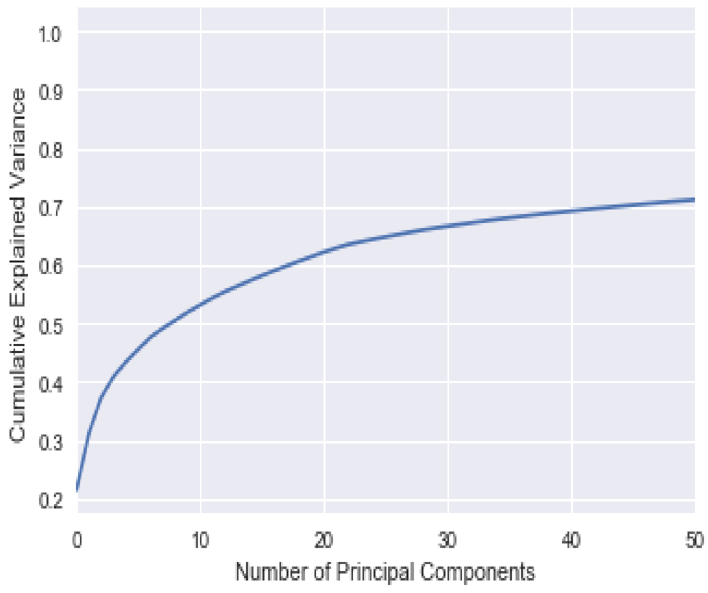

Fig. 3. The cumulative explained variance for the first 50 principal components.

either training data or testing data. Whilst these 50 features no longer have any real world meaning, each instance still represents a 5 second signal segment of a recorded activity.

The 10-fold cross validation technique is used in the training and testing phase for each classification model. Using the validation technique, the 9120 instances are divided into 10 partitions of 912 instances. The selected instances in each partition is decided randomly. The cross-validation process is repeated 10 times: each time a single partition is used as the training data for the model and the other 9 partitions are used as testing data. Each time this process is conducted a partition which has not yet been selected for training is selected. Each repeat has its own F1 score which represents how accurate that classification was, the cross-validation accuracy score is the average of these. This validation method makes use of all the data for both training and test purposes, which is beneficial since it means no data has been wasted when developing the models.

The mean F1 accuracy scores for the classification techniques are shown in Table II alongside their 95% confidence interval. The most accurate classification technique is the SVM model using a linear kernel which is expected to be 97% accurate when classifying new data. However, the SVM model using a Gaussian kernel is the worst classifier with only 70% expected accuracy. The decision tree and KNN models perform only slightly worse than the linear kernel SVM with expected accuracys 94% and 96% retrospectively.

Heatmaps for the confusion matrices for the SVM (linear kernel), KNN and decision tree models are shown in Figure 4, Figure 5 and Figure 6 retrospectively. The most common case of misclassification for each model is that an instance of jumping is misclassified as an instance of rowing. The misclassification occurs 77 times when using the SVM model, 56 times for the decision tree and 142 times using the KNN model. Both the SVM and decision tree model also misclassify a significant amount of 5 second segments as jumping instead of rowing. The next most common misclassification for the models was caused by confusing the activities jumping/ascending stairs and 'Walking on a treadmill with a speed of 4 km/h (15-degree inclined position)'. Whilst this misclassification happens both ways for the SVM and decision tree, the reverse misclassification does not occur for the KNN model.

Table III shows the run time for training and testing each model. The models were generated using a Juypter Notebook (Python 3), on a desktop computer with an Intel Core i5-4590 @ 3.30GHz and 8.00 GBs of RAM, running on a Windows 10 operating system. The SVM model takes the longest time to build and test (approximately 20 seconds longer than the other models), however it is also the model which provides the most expected accuracy.

TABLE II
THE MEAN F1 SCORES AND THEIR 95% CONFIDENCE INTERVAL

Technique	***F1 Score***	***95 % Confidence Interval***
SVM (Linear)	0.97	± 0.01
KNN	0.96	± 0.01
Decision Tree	0.94	± 0.01
SVM (Gaussian)	0.70	± 0.02

[a]Where $k = 6$ for KNN.

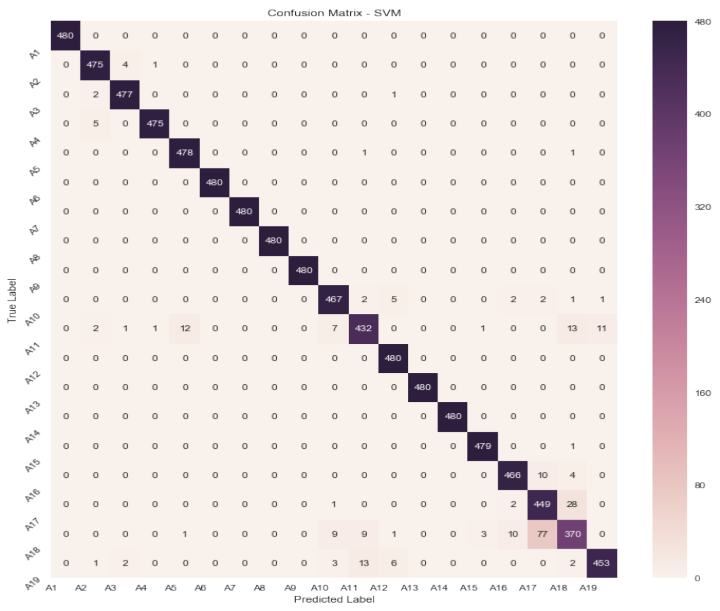

Fig. 4. Heatmap of the confusion matrix for the SVM (Linear) model.

VI. Discussion and Conclusions

The best model for this given classification problem is arguably either the SVM or KNN. The SVM (linear) model has a slightly higher correct classification rate, however training it

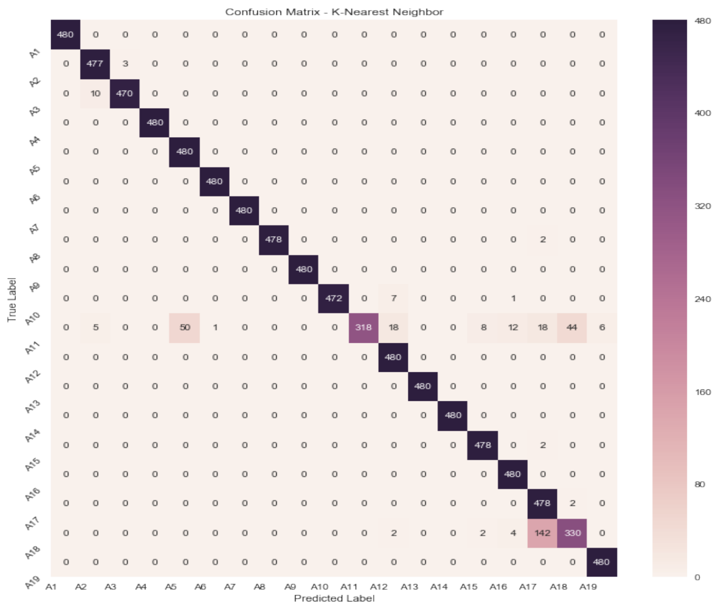

Fig. 5. Heatmap of the confusion matrix for the KNN model.

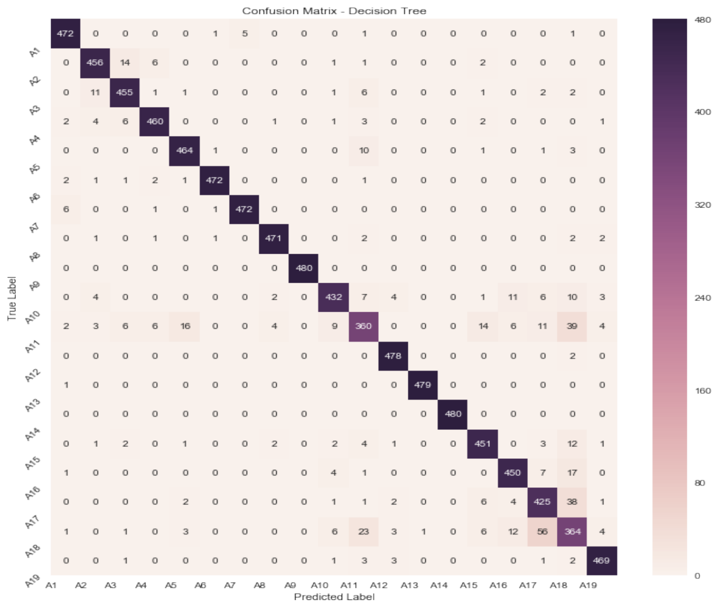

Fig. 6. Heatmap of the confusion matrix for the decision tree model.

TABLE III
THE RUN TIME FOR TRAINING AND TESTING EACH MODEL

Technique	***Training and Testing Run Time***
SVM (Linear)	19.40416518669943
KNN	2.7343802393945396
Decision Tree	4.795895543704778

requires considerable more time than the other models. However, the KNN model has a slightly worse correct classification rate but requires much less training time, although the time for predicting a new instance will be greater than other models since it is a lazy learner model and must calculate the Euclidian distance from the new instance to all the training instances. The decision tree is a satisfactory model for the classification problem with its correct classification rate only trailing slightly behind the KKK and SVM model. However, since decision trees are prone to overfitting it should be the least favoured model without significant pruning. Although it is important to note that by applying such significant dimensionality reduction via PCA, the risk of overfitting will have been significantly reduced.

Barshan's Comparative study on classifying this data set used PCA after statically manipulating the data set. The statistical manipulation required expert knowledge on how the data was obtained and each features true meaning. After significantly reducing the features from a set of size 5625 to 1170, PCA was used to select the first 30 principal components. This paper however explores the dataset without the expert knowledge that Barshan possessed due to collecting the data. The size of the data set is instead reduced using PCA immediately and the first 50 principle components which provide over 70% of the variance in the data are selected. The first 350 principle components explain roughly 90% of the variance, however such a large quantity of features would result in much larger training times for the models.

This paper shows that classification techniques can be very effective on this data set, even without statistically manipulating the data during pre-processing. The rule based algorithm in the Barshan study has a much lower correct classification rate (84.5%), compared to this decision tree model demonstrated in this paper (94%). This papers SVM model performs similar with regards to correct classification rate, 97% compared to the Barshan study model 98.8%. As does the KNN algorithm, 96% compared to 98.7%. Future research could implement different feature reduction techniques to further try to maximize these classification rate.

In this paper we have discussed and compared the application of various Machine Learning classification techniques to the same data set. We evaluated the successfulness of these techniques when used on this data set, by comparing their correct classification rates and the time required to train and test the model using 10-fold cross validation. The SVM (linear) and KNN (k=6) models are the best models which were tested. If the long pre-processing time required to train an SVM model is not an issue, then this model should be used to classify new data. If the pre-processing time is an issue, then the KNN is the next most suitable model. Finally, if the pre-processing time for training and the processing time requrired for predictions needs to be short, then the decision tree model is a suitable model choice. It should be noted that the classifications can only predict activity based on the data, if two activities are semantically very different, but the data for the classification is similar, misclassifications can still occur.

Which is why the data set manage to misclassify activities like rowing as jumping, whilst the same activities would be easy to differentiate visually.
As this paper shows, Machine Learning techniques can quite clearly be successful when it comes to classifying human activity using sensors, even if the various classifications are as similar as standing in different environments. Due to the large amount of potential applications of such sensors and the large amount of possible human test subjects (such as elderly seeking health care), there is a great variety of research directions that can be explored. For instance, consumer price Virtual Reality (VR) headsets are slowly being adopted [17]. VR software developers could benefit from Machine Learning techniques being applied to the motion sensor data retrieved from users of VR hardware. The data retrieved could be cross referenced with a large set of training data that classifies weather a user is comfortable when rotating their head in certain ways. However, the retrieval of such data is still limited to those willing to give consent for their individual data to be used and analysed.

VII. Appendix

Google drive link - contains python code used to pre-process the data and apply the classification techniques, alongside the original data:
https://drive.google.com/drive/folders/10R3O9EdBmaFiMY7XITkj18fVV27b8KPs?usp=sharing

References

[1] P. Turaga, R. Chellappa, V. S. Subrahmanian, and O. Udrea, "Machine recognition of human activities: A survey," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 18, no. 11, pp. 1473–1488, 2008.

[2] R. Bogue, "Mems sensors: past, present and future," *Sensor Review*, vol. 27, no. 1, pp. 7–13, 2007.

[3] R. Maboudian, W. R. Ashurst, and C. Carraro, "Self-assembled monolayers as anti-stiction coatings for mems: characteristics and recent developments," *Sensors and Actuators A: Physical*, vol. 82, no. 1, pp. 219–223, 2000.

[4] G. Ciuti, L. Ricotti, A. Menciassi, and P. Dario, "Mems sensor technologies for human centred applications in healthcare, physical activities, safety and environmental sensing: a review on research activities in italy," *Sensors*, vol. 15, no. 3, pp. 6441–6468, 2015.

[5] H. Alemdar and C. Ersoy, "Wireless sensor networks for healthcare: A survey," *Computer Networks*, vol. 54, no. 15, pp. 2688–2710, 2010.

[6] I. Laptev, M. Marszalek, C. Schmid, and B. Rozenfeld, "Learning realistic human actions from movies," in *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*. IEEE, 2008, pp. 1–8.

[7] D. Chen, Y. Chang, R. Yan, and J. Yang, "Tools for protecting the privacy of specific individuals in video," *EURASIP Journal on Applied Signal Processing*, vol. 2007, no. 1, pp. 107–107, 2007.

[8] F. Dufaux and T. Ebrahimi, "Scrambling for privacy protection in video surveillance systems," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 18, no. 8, pp. 1168–1174, 2008.

[9] J. R. Kwapisz, G. M. Weiss, and S. A. Moore, "Activity recognition using cell phone accelerometers," *ACM SigKDD Explorations Newsletter*, vol. 12, no. 2, pp. 74–82, 2011.

[10] K. Altun, B. Barshan, and O. Tunçel, "Comparative study on classifying human activities with miniature inertial and magnetic sensors," *Pattern Recognition*, vol. 43, no. 10, pp. 3605–3620, 2010.

[11] B. Barshan and M. C. Yüksek, "Recognizing daily and sports activities in two open source machine learning environments using body-worn sensor units," *The Computer Journal*, vol. 57, no. 11, pp. 1649–1667, 2013.

[12] K. Altun and B. Barshan, "Human activity recognition using inertial/magnetic sensor units," in *International Workshop on Human Behavior Understanding*. Springer, 2010, pp. 38–51.

[13] C. Z. Janikow, "Fuzzy decision trees: issues and methods," *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 28, no. 1, pp. 1–14, 1998.

[14] L. E. Peterson, "K-nearest neighbor," *Scholarpedia*, vol. 4, no. 2, p. 1883, 2009.

[15] M. A. Hearst, S. T. Dumais, E. Osuna, J. Platt, and B. Scholkopf, "Support vector machines," *IEEE Intelligent Systems and their applications*, vol. 13, no. 4, pp. 18–28, 1998.

[16] S. Wold, K. Esbensen, and P. Geladi, "Principal component analysis," *Chemometrics and intelligent laboratory systems*, vol. 2, no. 1-3, pp. 37–52, 1987.

[17] P. R. Desai, P. N. Desai, K. D. Ajmera, and K. Mehta, "A review paper on oculus rift-a virtual reality headset," *arXiv preprint arXiv:1408.1173*, 2014.