UNIVERSITY OF OKLAHOMA

GRADUATE COLLEGE

A COMPARISON OF MACHINE LEARNING GESTURE RECOGNITION TECHNIQUES FOR MEDICATION ADHERENCE

A THESIS

SUBMITTED TO THE GRADUATE FACULTY

in partial fulfillment of the requirements for the

Degree of

MASTER OF SCIENCE IN ELECTRICAL AND COMPUTER ENGINEERING

By

Joseph Sullivan

Norman, Oklahoma

2020

A COMPARISON OF MACHINE LEARNING GESTURE RECOGNITION TECHNIQUES FOR MEDICATION ADHERENCE

A THESIS APPROVED FOR THE

DEPARTMENT OF ELECTRICAL AND COMPUTER ENGINEERING

BY THE COMMITTEE CONSISTING OF

Acknowledgment

The data in this study was graciously made available for use by Dr. Hazem Refai's 2017 electrical engineering senior capstone groups at the University of Oklahoma.

Thank you to Summer Reif, Badre Ayadi, Saad Ayadi, Nathan Pierce, and Charles Attey on Team Green [1] and Andrew Meyer, Ian Grady, Garret Robinson, Alexander Horton, and Paulius Velesko on Team Blue [2].

Table of Contents

List of Figures

List of Tables

Abstract

Every year, many poor health outcomes are the result of patients missing their medication, as prescribed by their healthcare providers. Guidance and reminders to these patients would result in better health outcomes and significant financial savings to the economy. This thesis utilizes accelerometers and gyroscopes, which are widely available inside devices (e.g., smart phones and watches) to actively monitor patient activities, including those related to adherence to medication regimens. Different machine learning techniques are compared for recognizing when a pill bottle has been opened. Such actions could remind the patient to take their medication if an opening were not detected. An artificial neural network (ANN) model will be compared with a support vector machine (SVM) and a K-nearest neighbor (KNN) classifier. The models are trained on data collected by former University of Oklahoma students. Raw (normalized) sensor data is used, without extensive data processing or feature extraction. A neural network proves the most promising with an accuracy of 98.12%, as well as the greatest flexibility in data pre-processing requirements. KNN achieved high accuracy, although results were likely due to overfitting limited data with the simple model. SVM did not perform as well as the others, however; it did achieve similar results to previous research utilizing the approach (e.g., ~95% accuracy). Data collected from a greater number of gestures and additional test subjects is needed to verify generalization. A medication adherence system utilizing the developed model would be an acceptable approach.

1. Introduction

The low costs and small sizes of modern digital technology have allowed for a proliferation of smart devices. More devices are in the hands of people and more data is available than ever before. One approach, termed Internet of Things (IoT), aims to connect these embedded devices to send and receive data automatically, allowing us to improve quality of life by focused application of the data. IoT has been used extensively in the medical field, ranging from simple self-monitoring of sleep and heart rate to weight sensing and redistribution for overweight patients to prevent ulcers, and from monitoring vital signs in hospitals and alerting nurses to identifying problems arising in patient care.

Many IoT devices are equipped with accelerometers and gyroscopes, facilitating the measurement of 3-dimmensional linear accelerations and angular velocities. The applications of these sensors are many, but here we consider the use of the sensors to recognize specific human movements, such as smoking and opening a bottle of medication, to change behaviors leading to adverse health outcomes. Human motion recognition is an active area of research with many challenges. Dynamic variation occurs within each movement, allowing for slow or fast, as well as exaggerated or subtle movements. Another challenge is generalizing the test-user's data to multi-users who may have slight differences in their individual movements. Several processing steps are typically needed before the data can be used effectively.

A major area of concern in healthcare is medication adherence. Of all medication-related hospital admissions in the United States, 33 to 69 percent [3] are due to poor medication adherence, totaling more than $100 billion annually in increased medical costs.[4] Since pharmaceutical treatments are critical in today's healthcare system, adhering to the medication as prescribed is very important. A myriad of factors contribute to this problem—from suboptimal health literacy and lack of patient involvement in the decision-making process to absence of social/economical support, as well as complex prescription regimens and communication barriers, among many other reasons. The proliferation of IoT devices allows providers to intervene automatically, sending reminders to patients that could benefit from notifications.

This thesis aims to solve this problem by reporting on the use of a wrist-worn inertial sensor that recognizes when a bottle of medication has been opened. Sensing could eventually be developed further into a functionality that utilizes the inertial sensors commonly available in smart watches to automatically send reminders to patients when medication opening is not detected.

2. Background

2.1 Medical Applications of IoT devices

IoT devices are a growing industry in healthcare. Their proliferation has facilitated a shift in healthcare paradigms away from conventional hub-based services and towards a more personalized approach. Figure 1 demonstrates a system level view for personalized healthcare utilizing IoT.

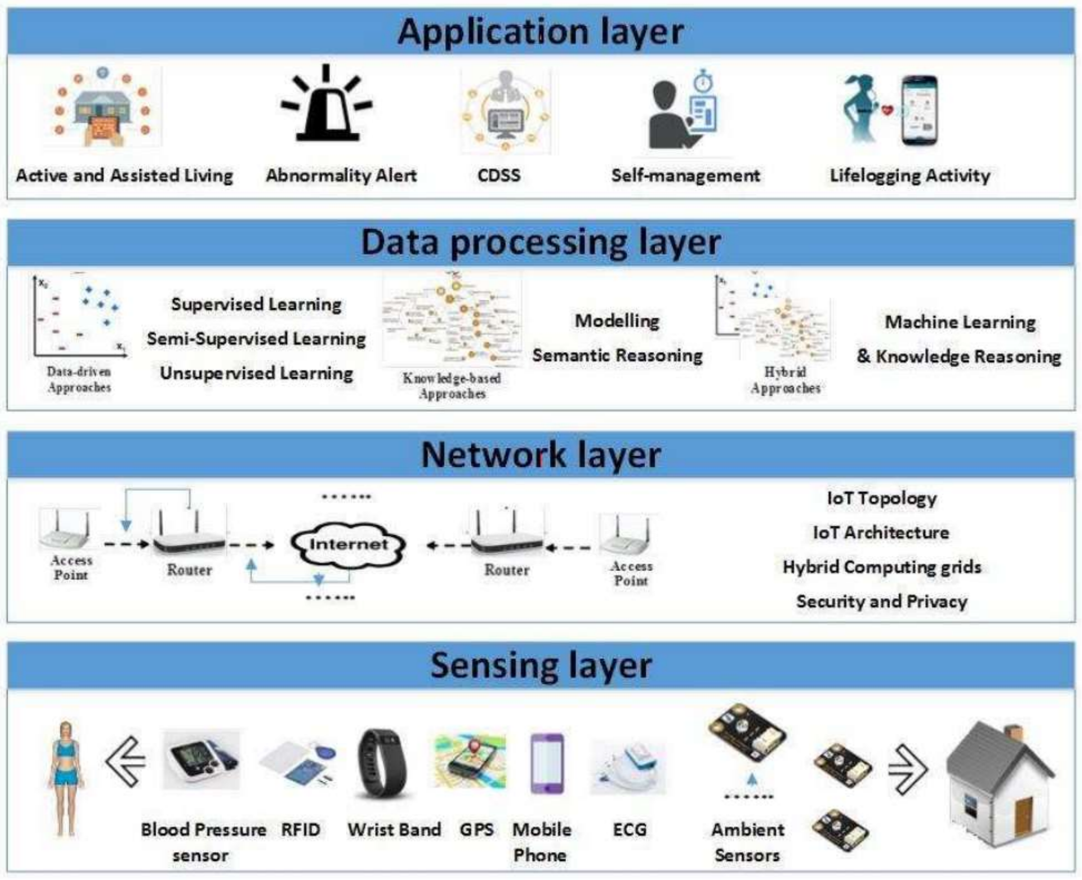

Figure 1. Service-oriented architecture layers for personalized healthcare [5].

With the ubiquity of IoT devices, more healthcare-relevant data is available than ever before. Wearable devices commonly come equipped with accelerometers, gyroscopes, barometric, and magnetic field sensors; these technologies enable the collection of linear accelerations, angular rotational velocities, altitude, and location with higher spatial resolution, respectively. More advanced wearables (e.g., physiological sensors like blood pressure cuffs, electrocardiograms, spirometers [air flow rate and lung volume], and electrooculography [eye movement tracking]) are available, although they are usually limited to use in healthcare settings. Other devices are deployed, rather than worn, to collect ambient data in a healthcare setting (e.g., thermometers, hygrometers, window and door sensors, light switches, Zigbee for location data, and RFID or NFC tags for object interaction sensing) [5].

Data processing is required to glean useful information. Machine learning is commonly used to improve some aspect of a patient's health outcomes. Data-driven approaches include supervised and unsupervised learning techniques. In supervised learning, the machine learns a complex function for mapping data as an input to a specific output based on training data; this requires labeled outputs in the data. Supervised learning techniques include, but are not limited to, artificial neural networks (ANN), support vector machines (SVM), K-nearest neighbor (KNN) classifiers, decision trees, hidden Markov models (HMM), and Gaussian mixture models (GMM). In unsupervised learning, the computer draws inferences from input data with no labeled outputs. Common unsupervised learning techniques include, but are not limited to, clustering techniques, such as K-means and

DBSCAN, anomaly detection, and expectation maximization. This thesis focuses on supervised learning with ANN, SVM, and KNN.

Several approaches have attempted to address the problem of medication adherence. One attempt involves using a smart pill bottle/stand that provides alerts to notify the patient to take their medication [6]. Another approach utilizes a smart medication dispenser to administer the prescribed medication at a predetermined time [7], although the dispenser does not validate that the medication was administered. Several approaches attempt to use computer-vision to solve the adherence problem [8-10]; however, these approaches prove inflexible and require a patient to take their medication in the view of the camera. A multi-sensor system was proposed in [11] and consists of a motion sensor, a wearable device, and a bed sensor. Although the system performed well, its complexity makes the system difficult to set up and operate. Another approach utilizes a wrist-mounted inertial sensor to determine when a pill bottle is opened, along with a camera to assist in the data training phase [12]. This approach combines the flexibility of a single wearable device with the use of a camera for augmenting the training and increase accuracy. The researchers utilized an HMM to achieve some level of success after extensive processing.

This thesis reports the use of a wrist-mounted inertial sensor for determining when a pill bottle has been opened. Camera assisted training was forgone to increase flexibility. Machine learning classification techniques (e.g., ANN, SVM, and KNN classifiers) were used. Models are presented, and current research on gesture recognition is reviewed. Models are compared for accuracy and the ability to reject other motions. During training, 100 repetitions of the movement were tracked. Further development is necessary for real-time tracking capability. Preprocessing was minimal, and training was accomplished utilizing 100 repetitions, including gyroscope and accelerometer data as a 6-dimensional time series. For differentiation, compared movements include opening a pill bottle and smoking. The classification aspect, rather than the full medication adherence system, is the targeted focus of the investigation.

2.2 K-Nearest Neighbor

The first classifier used was KNN—a non-parametric learning algorithm, meaning no underlying assumptions are made about the distribution of data [13]. Although the method was first developed in the 1950s, widespread use occurred in the 1960s after increased computing power was available. Given today's computing power, this method is widely used for pattern recognition.

The KNN method is based on human learning by analogy, comparing data points with those that are in the immediate vicinity. Each data point resides in an n-dimensional space. KNN calculates distances and searches for the K-closest points in the n-dimensional sample space to determine similarity between data points. Many distance metrics, including Minkowski, Euclidean, and Manhattan, are available. This thesis considers Euclidean distances of two objects described by n attributes. The Euclidean distance between $X_1\ and\ X_2$ points is defined as

$$d(X_1, X_2) = \sqrt{\sum_{i=1}^{n} (x_{1i} - x_{2i})^2} \tag{1}$$

The algorithm works by determining a test set of correctly labeled training data in the form $D = (X_i, y_i)$, where $X_i \in \mathbb{R}^n$ is the measured input data, and $y_i \in \mathbb{R}$ is the corresponding class label. For each point in test dataset T, distances are calculated to every point in the training set using (1). The closest K points for each test point are selected, and the most frequent class label among the K points is chosen as the estimate for the class label of the test point. When choosing between two classes, an odd K should be used to avoid a tie situation.

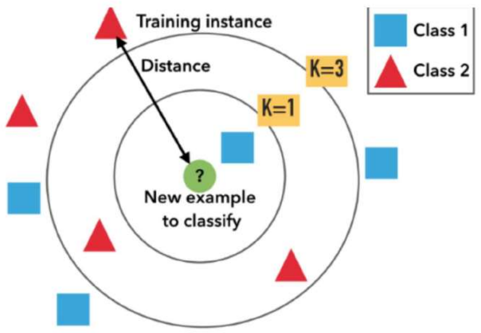

Figure 2. K-nearest neighbor algorithm visualization [30].

The KNN classifier is simple, yet powerful, requiring no training other than having correctly labeled available data. This simplicity can be a significant advantage, as training the data can be the most time-consuming activity of many classification techniques. However, simplicity comes at a cost when classifying new points, as the distance to every training point must be calculated. Hence, if $k = 1$ and D is a dataset with $|D|$ points, the number of computations required is $O(|D|)$. Utilizing search trees can decrease the number of computations to $O(\log|D|)$. Parallel implementation can further reduce running time to a constant, $O(1)$, which means number of calculations required is independent of the size of D.

2.3 Support Vector Machine

The next classification technique under review was SVM [13]. This technique was first presented in 1992 by Vladimir Vaprik et al., although the groundwork had been available since the 1960s. Training times for SVMs can be slow, albeit for many applications results are highly accurate because SVM can map complex nonlinear decision boundaries.

The idea behind any SVM is leveraging a nonlinear mapping to transform the dataset into higher dimensions. With this increased dimensionality comes increased separation between classes, when done correctly. SVM aims to find the hyperplane that separates the classes (i.e., a decision boundary). Support vectors (i.e., essential training points) and margins are used. Overfitting is much less likely to occur in SVMs when compared with other methods. The learned model can be compactly described using the support vectors. SVMs are commonly used in a variety of areas (e.g., handwritten digit recognition, object recognition, and speaker identification, as well as benchmark time-series prediction tests).

To illustrate how SVMs work, a linearly separable case is shown below in Figure 3.

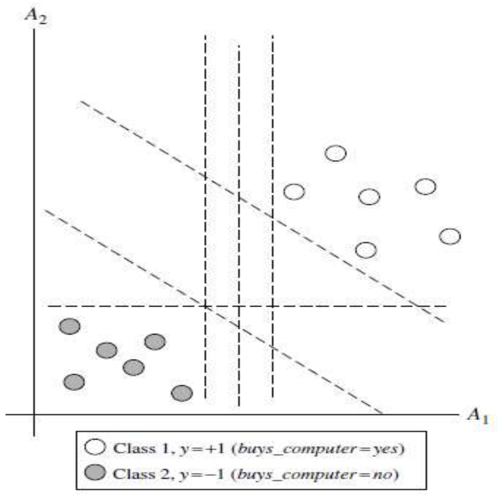
A_2
A_1
Class 1, y=+1 (*buys_computer=yes*)
Class 2, y=−1 (*buys_computer=no*)

Figure 3. Linearly separable training data set example [13].

In the figure above, an infinite number of possible linear decision boundaries are available. The task is choosing the best one. The line proves to be a plane or hyperplane in higher dimensions. To solve this problem, SVM searches for the maximum margin hyperplane. Figure 4 demonstrates the concept.

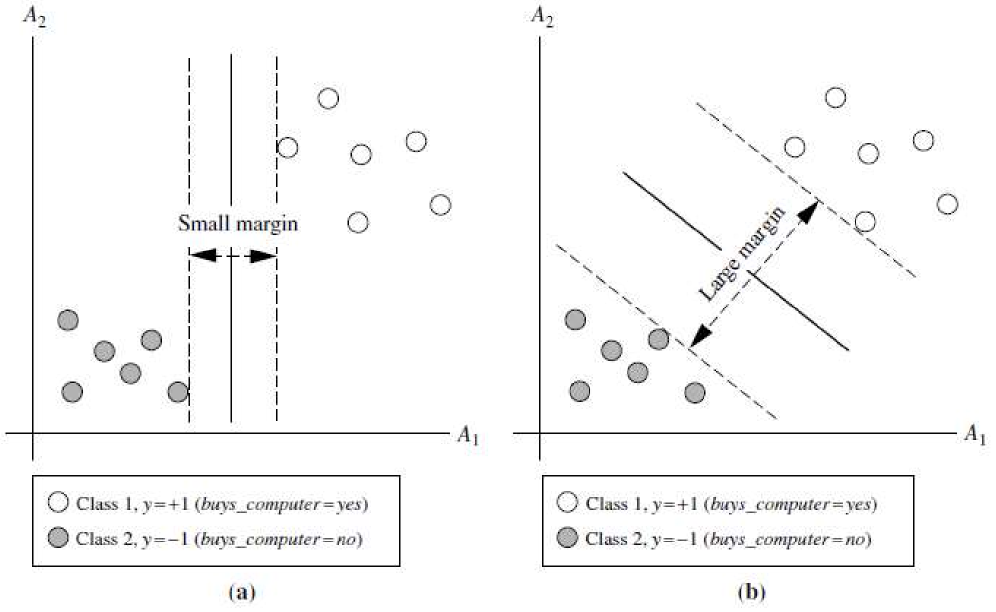
A_2
Small margin
A_1
Class 1, y=+1 (*buys_computer=yes*)
Class 2, y=−1 (*buys_computer=no*)
(a)
A_2
Large margin
A_1
Class 1, y=+1 (*buys_computer=yes*)
Class 2, y=−1 (*buys_computer=no*)
(b)

Figure 4. Two possible decision boundaries and their associated margins [13].

The figure above demonstrates that the margin is the distance orthogonal to the decision boundary to the closest points of the two classes. Mathematically, the decision boundary can be written as

$$\boldsymbol{W} \cdot \boldsymbol{X} + b = 0 \tag{2}$$

where $\boldsymbol{W} \in \mathbb{R}^n$ is a weight vector, and b is a scalar bias. In two dimensions, this reduces to

$$w_1 x_1 + w_2 x_2 + b = 0 \tag{3}$$

The sides of the margins can then be defined by

$$y_i = \begin{cases} +1, & w_1 x_1 + w_2 x_2 + b \geq 1 \\ -1, & w_1 x_1 + w_2 x_2 + b \leq -1 \end{cases} \tag{4}$$

The test points above or below the boundaries are sorted into the respective classes. The distance from the decision boundary to either margin hyperplane is $\frac{1}{||\boldsymbol{W}||}$. Because both margin hyperplanes are equidistant to the decision boundary, total width of the margin is $\frac{2}{||\boldsymbol{W}||}$. Support vectors are the points in the training data set that lie on the margin hyperplane and can compactly describe the SVM. The decision boundary is orthogonal to the line connecting the support vectors, and the margin width is the magnitude of the distance between the points. To solve this problem and find the maximum margin hyperplane and support vectors, a Lagrangian formulation is used to convert the problem into a constrained convex quadratic optimization. Exact details for this process are beyond the scope of this work, although they can be found in [13].

When the training data is linearly inseparable, a nonlinear transformation into a higher dimensional space is required. Kernel functions are used for transforming the data. Commonly used kernel functions are polynomials, gaussian radial basis functions, and the sigmoid function. These nonlinear SVMs compute the same decision boundaries as corresponding neural network classifiers. For instance, an SVM with a Gaussian radial basis function (RBF) gives the same decision hyperplane as a type of neural network known as a radial basis function network. An SVM with a sigmoid kernel is equivalent to a simple two-layer neural network known as a multilayer perceptron (with no hidden layers).

There are no general rules for choosing a kernel function to maximize the accuracy of the classifier. Typically, the choice does not make a significant difference. The training process for SVMs means that the SVM always finds the global minimum.

2.4 Artificial Neural Networks

Neural networks [13] are simply a set of connected input-output units called neurons, each with their own weight and activation function. A neuron collection connected in complex ways can "learn" by adjusting the weights of each neuron connection to model a complicated nonlinear function, mapping input data to output data. Neural networks were originally the domain of psychologists and neurobiologists and served as a tool for comparing a computational analogy to a real neuron. Recently, however, the field has proliferated in the data science community. Neural networks typically require large data sets and long training times, which can limit the scope of their applications. When feasible, though, few machine learning techniques are as flexible or powerful.

Several other advantages to neural network classifiers include a high tolerance to noise in the training data, as well as the ability to classify novel patterns. Also, very little knowledge on relationships between classes and attributes is needed, as relationships are automatically trained into the network. Neural networks are much better suited for continuous valued data when compared with other classifiers, such as decision trees. They have seen a wide variety of real-world applications, from recognizing handwritten characters to pathology and diagnostics in medicine, as well as for training computers to pronounce English words more naturally. Due to neural network's inherent parallel structure, the training process time can be drastically decreased by utilizing parallelization and GPUs. Recently, new techniques have been developed to extract rules from trained neural networks, allowing us to gain new insights into the situation, as well as to determine exactly what neural networks learn to recognize. An example of a fully connected neural network is shown below in Figure 5.

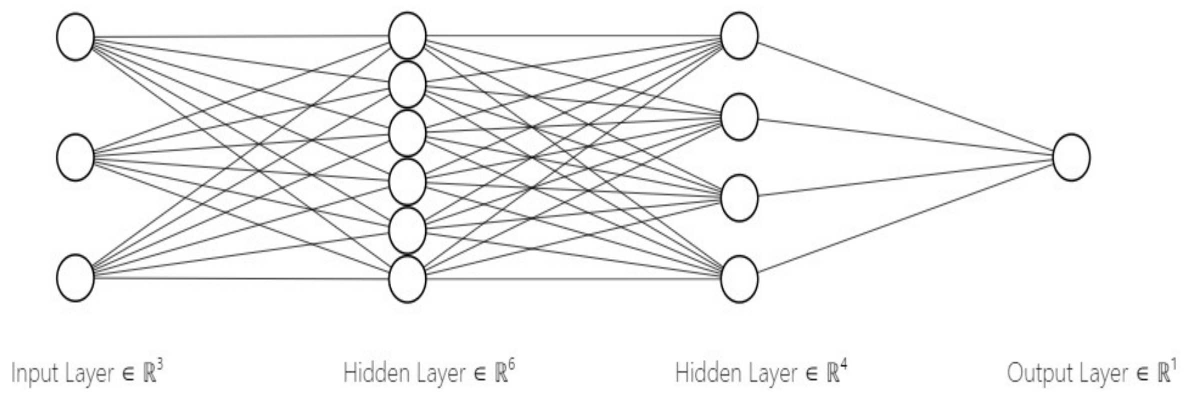
Input Layer $\in \mathbb{R}^3$
Hidden Layer $\in \mathbb{R}^6$
Hidden Layer $\in \mathbb{R}^4$
Output Layer $\in \mathbb{R}^1$

Figure 5. Neural network fully connected architecture.

In the example above, there are three inputs, six neurons in the first hidden layer, four neurons in the second hidden layer, and a single output neuron to determine the output class. Several different neural network architectures and training algorithms exist. This thesis focuses on a fully connected multi-layer feedforward network. The most popular network training algorithm is backpropagation. At first, network weights initialize to random values. Backpropagation forces the network to learn by iteratively processing data points in the training data, and then comparing the output to the true class. The algorithm adjusts neuron weights after each data point to decrease the mean-squared error of the prediction. Modifications are made starting with the output layer and working backwards. Convergence is not guaranteed, although weights, in general, will eventually converge.

Given a neuron, j, located in a hidden or output layer, the net input, I_j, to neuron j is given by

$$I_j = \sum_i w_{ij} O_i + \theta_j \tag{5}$$

where w_{ij} is the weight of the connection to the ith neuron on the preceding layer; O_i is the output of the ith neuron in the preceding layer; and θ_j is the neuron bias. The bias term acts as a threshold for varying neuron activation. Each neuron takes the net input from all neurons in the preceding layer, and then applies an activation function. Many different activation functions are used, with the most common being sigmoid, hyperbolic tangent, or rectified linear unit (ReLU).

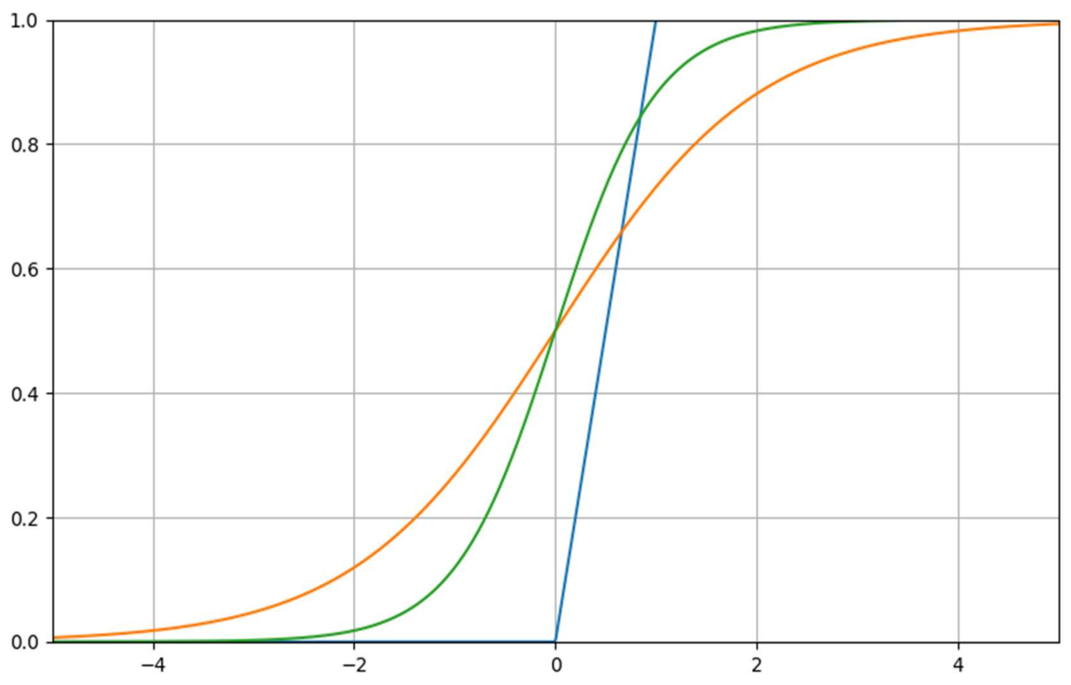

Figure 6. Common activation functions.

Error is propagated backwards by updating weights and biases in each layer as you move backwards to reflect the error of the network's prediction.

For neuron j in the output layer, error is determined by the equation

$$e_j = O_j(1 - O_j)(T_j - O_j) \tag{6}$$

where O_j is the actual output of the neuron, and T_j is the known true class of the training point. One should note that $O_j(1 - O_j)$ is the derivative of the sigmoid function.

For a neuron in the hidden layers, error of a neuron j is given as

$$e_j = O_j(1 - O_j)\sum_k e_k w_{jk} \tag{7}$$

where w_{jk} is the weight of the connection from neuron j in the current layer to neuron k in the next highest layer, and e_k is the error of neuron k.

For each training point, weights are updated using the following error values:

$$w_{ij} = w_{ij} + \lambda e_j O_i \tag{8}$$

where λ is the variable known as the learning rate, which is typically between 0 and 1. Backpropagation works by utilizing a gradient descent to minimize mean squared error of the network's prediction. This method can get stuck in local minima, which is why a variable learning rate is utilized. A too small learning rate leads to unnecessarily long training, while too large of a learning rate can lead to oscillations.

The biases are updated each iteration according to

$$\theta_j = \theta_j + \lambda e_j \tag{9}$$

Network training stops when the weights change less than given threshold, the accuracy reaches a set value, or a set number of epochs has expired.

2.5 Related Work

Gesture recognition has seen a recent proliferation of research. This thesis utilized 3D gyroscope and accelerometer-based gesture recognition. Three main strategies have been proposed by the research community: 1) probabilistic signal modeling, 2) temporal warping, and 3) machine learning-based classification.

The probabilistic gesture recognition approach has mainly utilized discrete [14-16] and continuous HMMs [17]. Kela et al. [16] use discrete HMMs (dHMM) from gesture velocity profiles. Raw data must be clustered first to reduce dimensionality and build a feature codebook. The second step consists of creating a discrete HMM using the sequences of vector codebook indexes. A correct recognition rate of 96.1% was obtained with 5 HMM states and a codebook from eight gestures performed by 37 people. In his approach to probabilistic gesture recognition, Pylvänäinen [17] proposed a system based on continuous HMM (cHMM) achieving a recognition rate of 96.76% on a dataset with 20 samples for 10 gestures performed by seven people.

The second approach is based on time warping samples using a set of reference gestures for guidance [18-20]. Liu et al. [19] presented a method using Dynamic Time Warping (DTW) from pre-processed signal data that offered gesture recognition and user identification rates of 93.5% and 88%, respectively.

The third strategy is based on machine learning-based classifiers [21-25]. Hoffman et al. [21] proposed a linear classifier and AdaBoost, resulting in a user-independent recognition rate of 98% for 13 gestures performed by 17 participants.

Sung-Jung Cho et al. [22] utilized a two-stage recognition algorithm consisting of a Bayesian belief network, followed by an SVM for confusing cases. Results showed a very high recognition rate, with 100 users the system achieving an average recognition rate of 96.9% on 11 gestures. This classifier worked on accelerometer data alone and was installed in Samsung cell phones, starting in 2005.

Wu et al. [23] proposed a classifier based on SVM. Each gesture was segmented in time, and then statistical measures (e.g., mean, energy, entropy, standard deviation, and correlation) were calculated for each time segment to synthesize final feature vectors. Resulting recognition rate was 95.21% for 12 gestures made by 10 individuals, which outperformed DTW results.

A recent study by Lefebvre et al. [24] proposed a method based on bidirectional long-short-term memory recurrent neural networks (BLSTM-RNN). Trained on data from 22 individuals performing 14 gestures, the BLSTM-RNN classifier was able to achieve a recognition rate of 95.57%.

A study by Stefan Duffner et al. [25] proposed a method based on a convolutional neural network (CNN) with a specific structure involving a combination of 1D convolution, averaging, and max-pooling operations. The work directly classified the fixed-length input matrix, composed of the normalized sensor data. The ConvNet proposed in [24] met or surpassed the methods in [21-23], showing promise for neural network use in gesture recognition.

CNN were also utilized by Sojeong Ha and Seungjin Choi [26] with data from 10 subjects performing 12 gestures, wearing three accelerometers (e.g., chest, right wrist, left ankle) and two gyroscopes (e.g., right wrist, left ankle). The novel CNNs recognized 91.94% of the test set. In this study, 10,000 gestures were used for training and demonstrated that in order for more gestures to be recognized, more training is required. Given the goal of picking out a single gesture, accuracy must be increased.

Casey Cole et al. [27] utilized an Apple Watch's accelerometer and an ANN for recognizing the gesture of smoking a cigarette at an accuracy of 85-95%. The goal of the study was providing a way to recognize smoking and intervene or reduce the number of cigarettes consumed per day. This investigation is similar to this thesis in that improving health outcomes was the primary motivator. With a similar goal of smoking cessation, Abhinav Parate et al. [28] used multiple IMUs, trajectory-based methods, and a probabilistic model to detect smoking gestures at 95.7% accuracy.

2.6 Thesis Contributions

This thesis compares SVM, KNN, and ANN classifiers to map out and classify gesture movements. The models in this study were built with the idea of minimizing the burden of data preprocessing and utilizing open-source python packages, with the neural network model using a dense, fully connected structure. Study data was limited to two gestures: smoking and opening a pill bottle. Each dataset consisted of 100 repeated gestures.

Data is linearly interpolated so that the datasets for the two gestures and the two data types, namely gyroscope and accelerometer, are equal in length. After interpolation, data was standardized to remove the mean and scale to unit variance. The input signal is not partitioned into smaller time segments or sequentially processed, like with HMMs [13- 16], time warping based methods [17- 19], or recurrent neural networks [22]. Models were trained on a randomly sampled set of 100 repeated gestures of equal length. Training on a large number of repeated gestures enables the models to effectively learn an individual's motions without complicated preprocessing or feature extraction.

In this study, features were automatically learned from the raw (i.e., standardized) input signal. Thus, no synthesized feature design, such as for HMM codebooks or statistical descriptors, are required. Limitations of this dataset prevent testing on a user independent basis. Because each gesture was performed by a different individual, unique differences in our dataset result in easily classifiable data, as well as an overestimation of recognition accuracy.

3. Methodology

3.1 Data Collection

Gestures collected included smoking and opening a pill bottle [1]. TI SensorTag IoT Demo Kit was used to collect the necessary data. This small, self-contained module was composed of multiple connectivity options and a variety of sensors. TI CC2650 SimpleLink SensorTag hardware performed data acquisition. SensorTag utilizes Bluetooth to communicate with a user's cell phone, and an app is used as the control interface.

Sensor default data sampling rate was 0.3 seconds with "wake on shake" enabled. To maximize data collection, the "wake on shake" feature was disabled, and data sampling rate was changed to 0.1 seconds. Data could be viewed in real time via the app and output to a data file.

The device contained a number of sensors: light, temperature, accelerometer, gyroscope, magnetometer, pressure, humidity, microphone, and magnetic. Of these, only the accelerometer and gyroscope were used in this study. Accelerometers are primary sensors for classifying movement and are extremely useful for indicating orientation (e.g., SensorTag accelerometer measures x, y, and z acceleration in G's). The SensorTag gyroscope measures rotation around the x, y, and z axes in degrees per second. Gyroscopic data was determined to be especially useful, given the rotational nature of opening pill bottles.

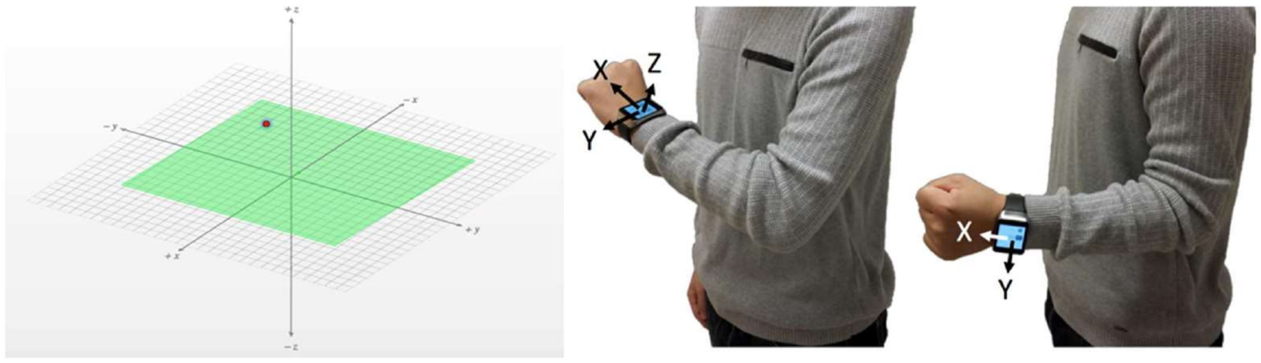

Figure 7. SensorTag Axes orientation; red dot indicates TI icon location [1].

Data was collected when test subjects performed 100 repetitions of the desired gesture at an accommodating repeatable pace. A TI SensorTag was placed on the wrist of each subject's hand that was performing the gesture. Each gesture was recorded as one individual's sensor data, with each gesture performed by a different person. The accelerometer and gyroscope data were collected at 10 Hz and saved to comma separated value (.csv) files for analysis.

3.2 Data Processing

Data labelled in .csv files were imported into python using the pandas data science package. The dataset for the pill bottle gesture contained 1047 x, y, and z accelerations and 1201 x, y, and z rotational velocities. The dataset for the smoking gesture contained 1771 x, y, and z accelerations and rotational velocities. Accelerations were recorded as multiples of G in the range [-1.0, 1.0]; rotational velocities were measured in degrees per second. The raw data is presented below as a histogram in Figures 8 thru 13.

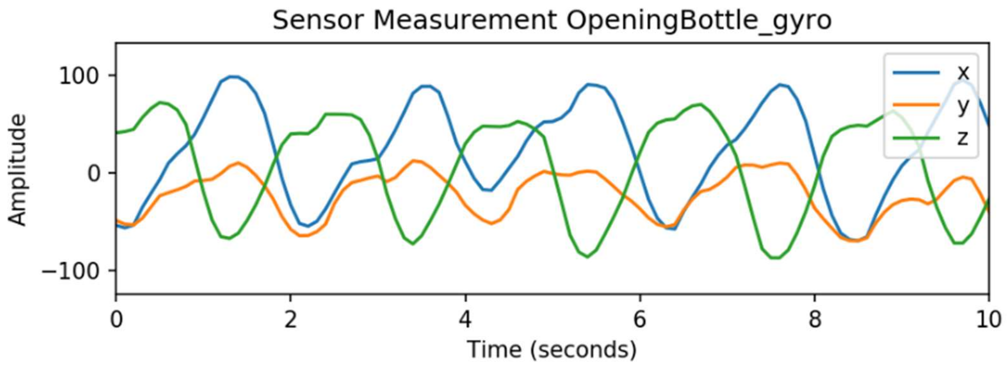

Figure 8. First 10 seconds of gyro data for bottle opening gesture.

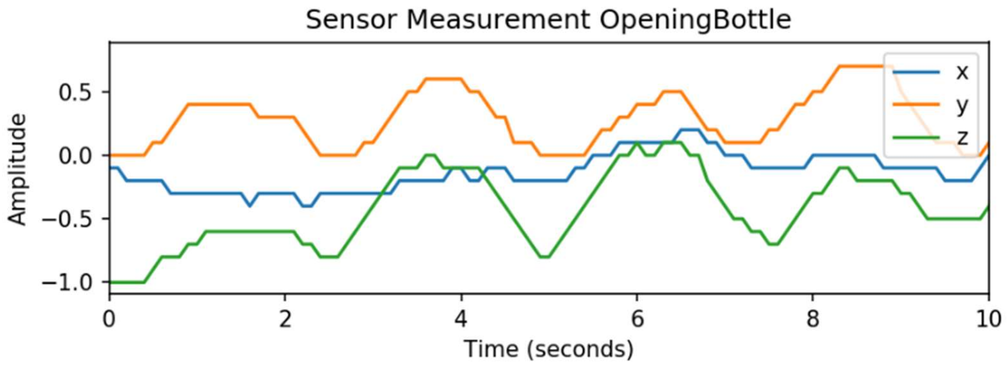

Figure 9. First 10 seconds of accelerometer data for bottle opening gesture.

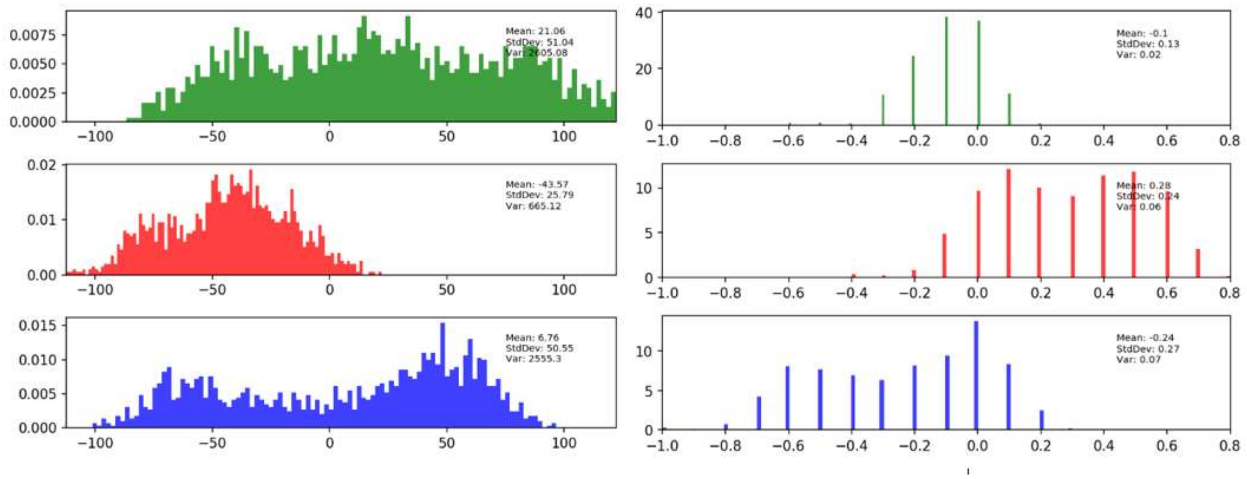

Figure 10. Data histogram (e.g., gyro left, accelerometer right) for bottle opening gesture.

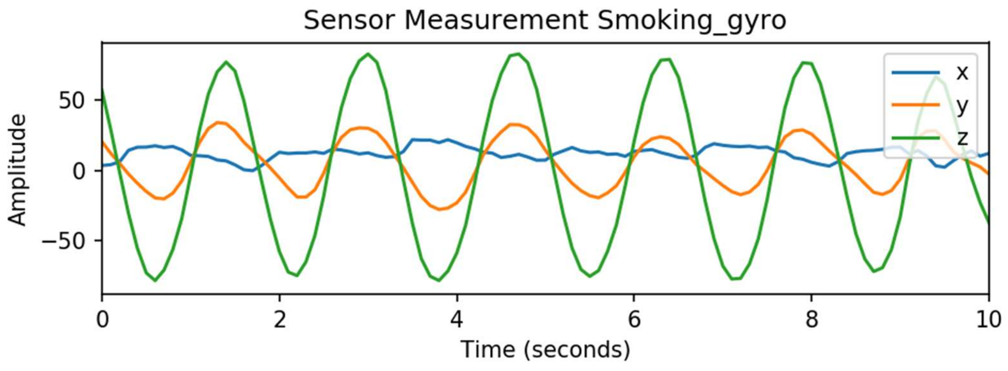

Figure 11. First 10 seconds of gyro data for smoking gesture.

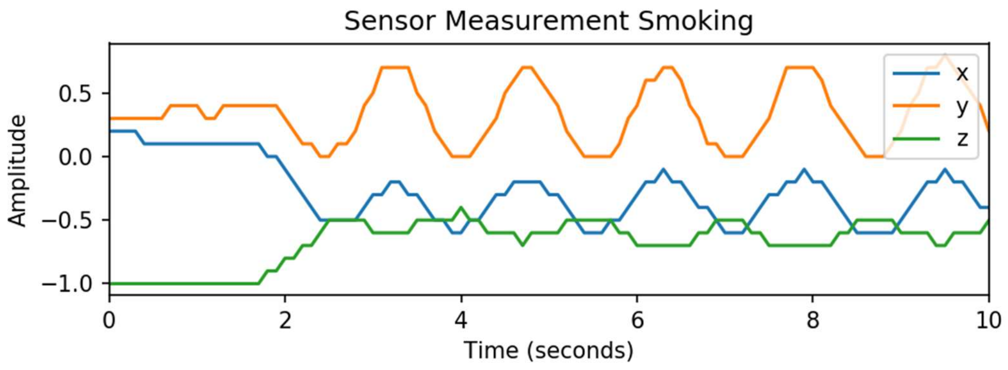

Figure 12. First 10 seconds of accelerometer data for smoking gesture.

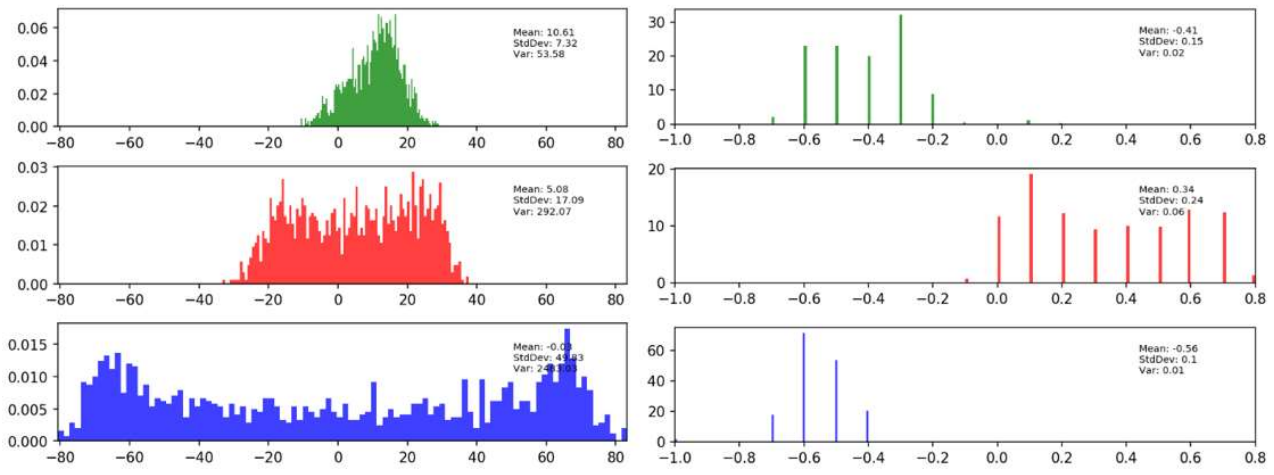

Figure 13: Data histogram (e.g., gyro left, accelerometer right) for smoking gesture.

The main difference between sensors is readily apparent in the dataset histograms. Accelerometer measurements can read only in 0.1 G's, which is much less precise than the gyroscope that measures to a sensitivity of 0.1 degrees.

Gyroscope data for the bottle opening gesture had a significantly larger spread in the X axis, slightly more spread in the Y axis, and roughly equal spread in the Z axis when compared with the smoking gesture. A larger spread was observed only in the Z direction for the accelerometer data.

A fast Fourier transform (FFT) was computed for the dataset to investigate noise level. FFT was then passed through a low-pass filter to eliminate any high frequency noise from the signal. An inverse FFT was computed to retrieve the filtered signal in the time domain. Figure 14 shows a selection of analysis results.

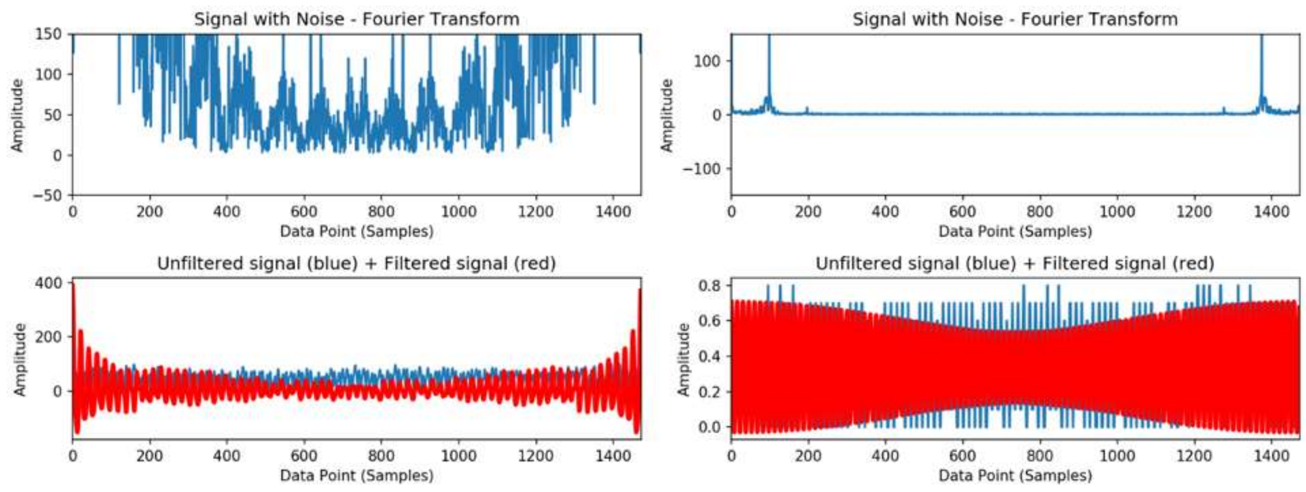

Figure 14. Fourier analysis for bottle opening gyro data in Z axis (See left graphs) and smoking accelerometer data in Y axis (See right graphs).

Fourier analysis results indicated that filtering was unnecessary, as the filter introduced unwanted distortion in the signal and a large high frequency noise component was not present. Next, the dataset was standardized to remove the mean and scale to unit variance. The mean of each attribute is given by equation 10, where k is the dimensional index; i is the sample index; and n is the number of samples.

$$\mu_k = \frac{1}{n}\sum_{i=1}^{n} x_{ki} \tag{10}$$

The standard deviation for each dimension is given by Equation 11:

$$\sigma_k = \sqrt{\frac{1}{n}\sum\nolimits_{i=1}^{n} (x_{ki} - \mu_k)^2} \tag{11}$$

Each data point had the dimensional mean subtracted, and the result was divided by the dimensional standard deviation to standardize the dataset.

$$\overline{x_{k\imath}} = \frac{x_{ki} - \mu_k}{\sigma_k} \tag{12}$$

Standardization was performed on an axis-by-axis basis. Each standardized sensor axis measurement had a mean of zero and a standard deviation of one.

After standardization, a principal component analysis (PCA) was performed on the data. First, a 2D PCA was run, which was followed by a 3D PCA. Principle component representation of the data is given below in Figure 15.

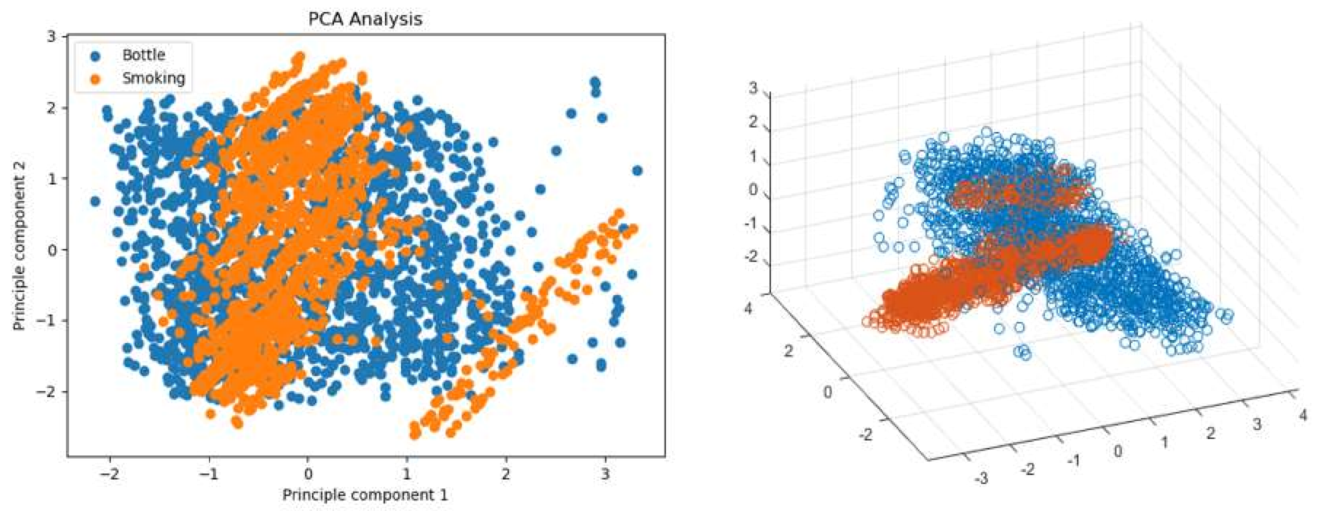

Figure 15. Dataset visualized by PCA.

The KNN classifier was used to determine the effect of the number of principle components on the classifier accuracy, which proved a way to measure interclass separation. The KNN model achieved accuracies of ~92% for n=2 and ~95% for n=3, as pictured above. Results for all six PCA possibilities are presented below in Figure 16.

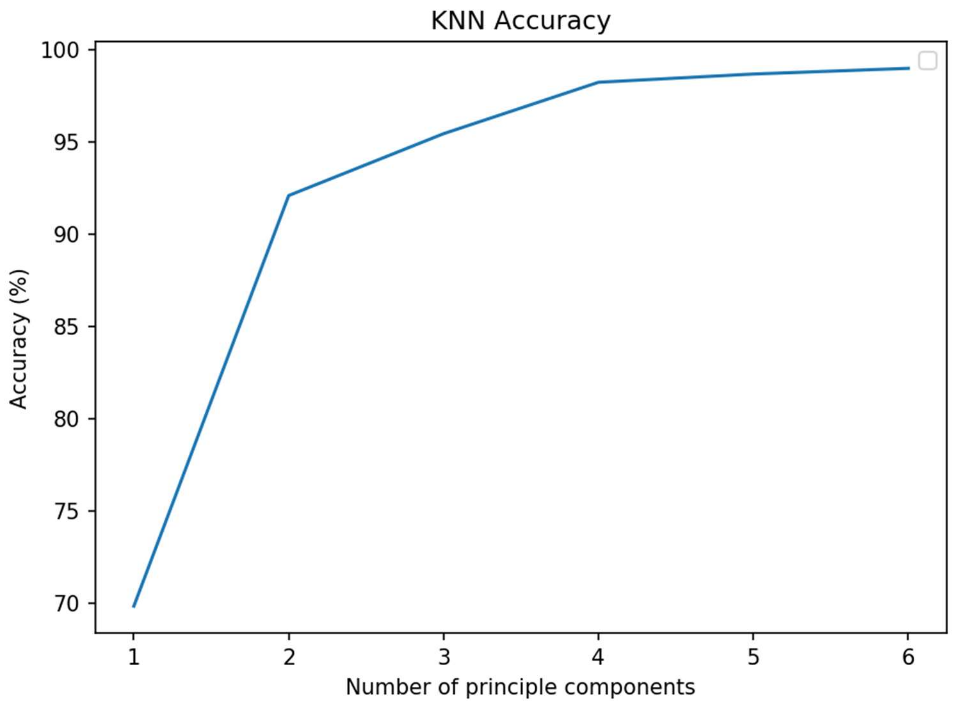

Figure 16. Model recognition accuracy vs. number of principle components included.

Overall, accuracy increased as the number of principal components increased, which was expected. This demonstrates an increase in inter-class separation in relation to intra-class separation as more principal components are included. Diminishing return on accuracy improvement was due to the fact that principle components are ordered according to their contribution to the total signal. Less information is added with the addition of each principle component. Nearly all information was added by n=4. A hypothesis to explain this behaviorthat the first three principle components were dedicated to the 3D motion; a fourth allowed a layer of redundancy, with further redundancy limiting its effect. In the end, PCA was not used to reduce the dimensionality of the dataset, as a 6D dataset is not computationally expensive to process.

3.3 Model Details

Three models, namely KNN, SVM, and ANN were used to classify gestures. Each model demonstrated unique advantages and disadvantages.

KNN requires only labeled data of equal length. Technically, no training occurs. At each new point tested, distances to each training point must be calculated and compared to select the K nearest. Although this method is simple, it can be a burden computationally given large amounts of data or neighbors.

The KNN model was first tested using 70% of the dataset for training and the remaining 30% for testing. Number of neighbors included in the calculation was varied. Results are plotted below in Figure 17.

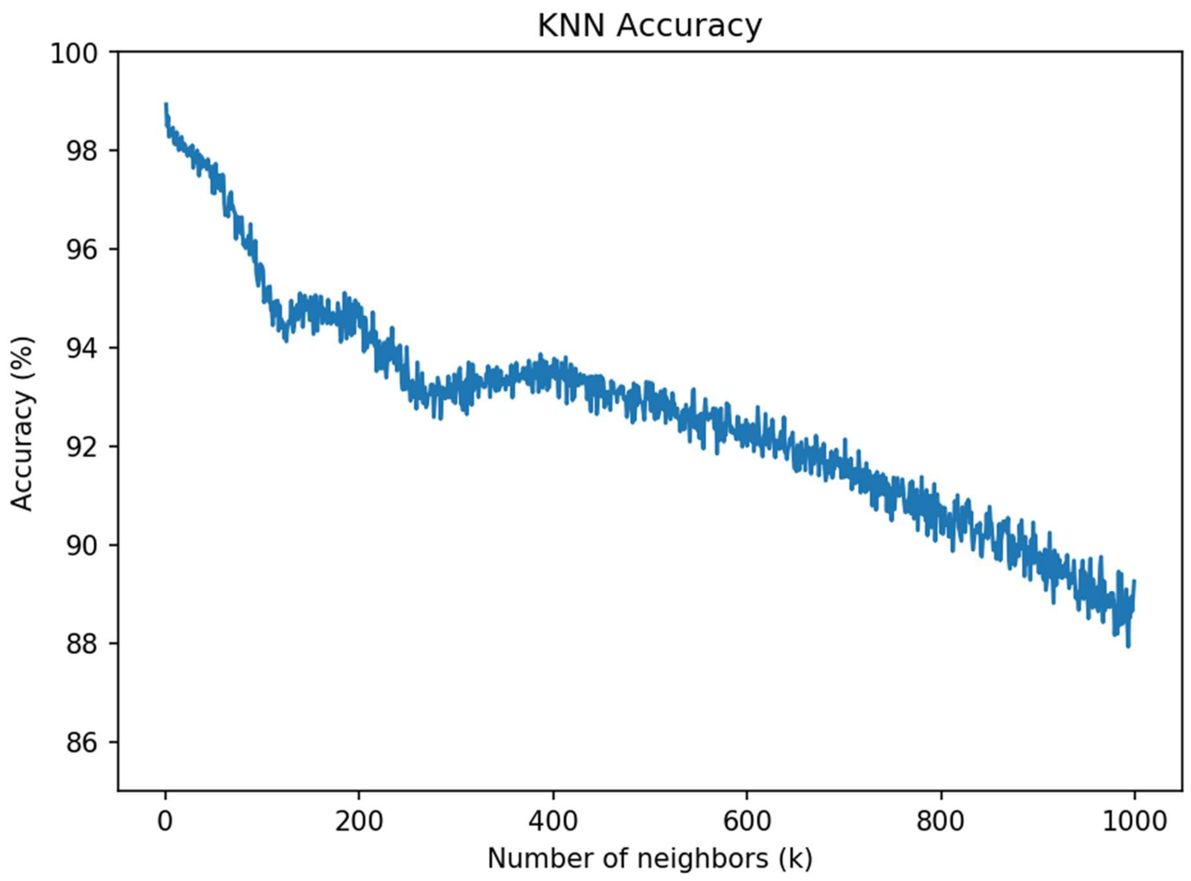

Figure 17. KNN accuracy as a function of neighbors included.

The plot shows that the highest accuracy was attained with the case K=1—the simplest nearest neighbor classifier. This is advantageous computationally, although results are skeptical due to a user set limited to two.

Next, accuracy was calculated for varying values of the test ratio—the portion of the dataset devoted to testing. Results are shown below in Figure 18. As the test ratio increases, less data is devoted to training; hence, accuracy is expected to decrease.

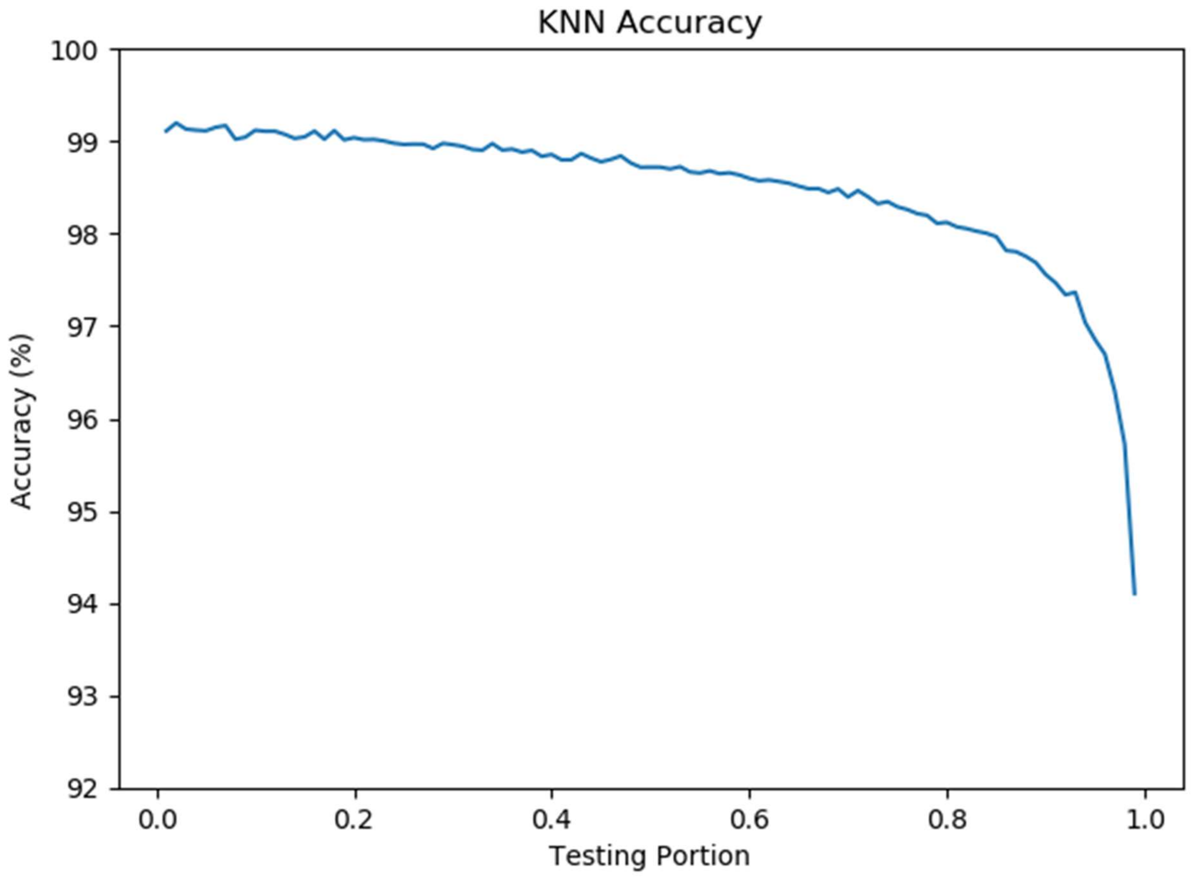

Figure 18. K-Nearest neighbors accuracy as a function of test ratio.

The KNN classifier holds steady near ~99% accuracy at low test ratios (i.e., high number of training points) and declines slowly (~98%) through the mid test ratios, with a precipitous drop at very high-test ratios, although the drop was only limited to 94%. The accuracy of this classifier indicates overfitting the dataset. More individuals should be tested to fully validate KNN for gesture recognition on normalized raw sensor data.

The SVM algorithm guarantees a "globally" optimal solution, given its hyperplane decision boundary. While this might not be *the globally optimal solution*, it should be noted that this method does not rely on random initializations—only random sampling for training. SVMs can take some time to train, but once trained, evaluations can quickly be performed.

First, four different kernel functions were tested: 1) the Gaussian radial basis function (RBF), 2) polynomial, 3) linear, and 4) sigmoid function. Accuracies of the different kernel functions were evaluated using 30% of the dataset for testing. Results are shown below in Figure 19.

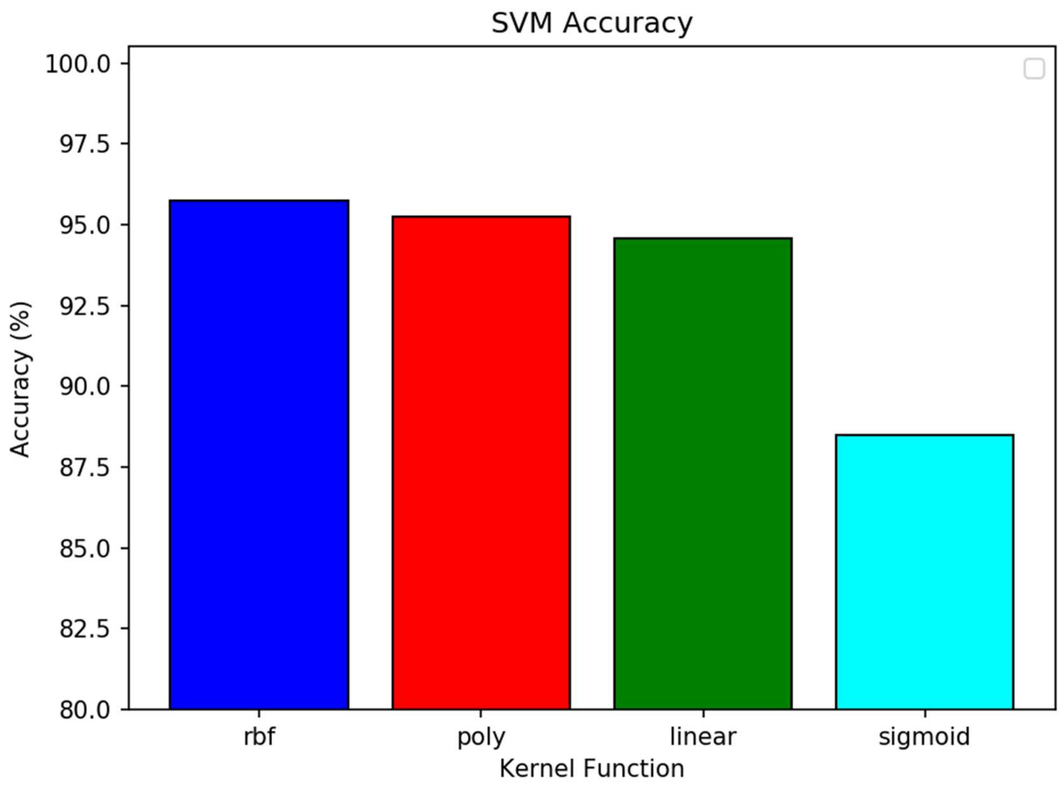

Figure 19. Gesture recognition accuracy for different kernel functions.

Through this analysis, the Gaussian radial basis function proved the best choice for the SVM kernel function. Next, accuracy was calculated for varying values of the test ratio, as was done above for the KNN classifier. Results are shown below in Figure 20.

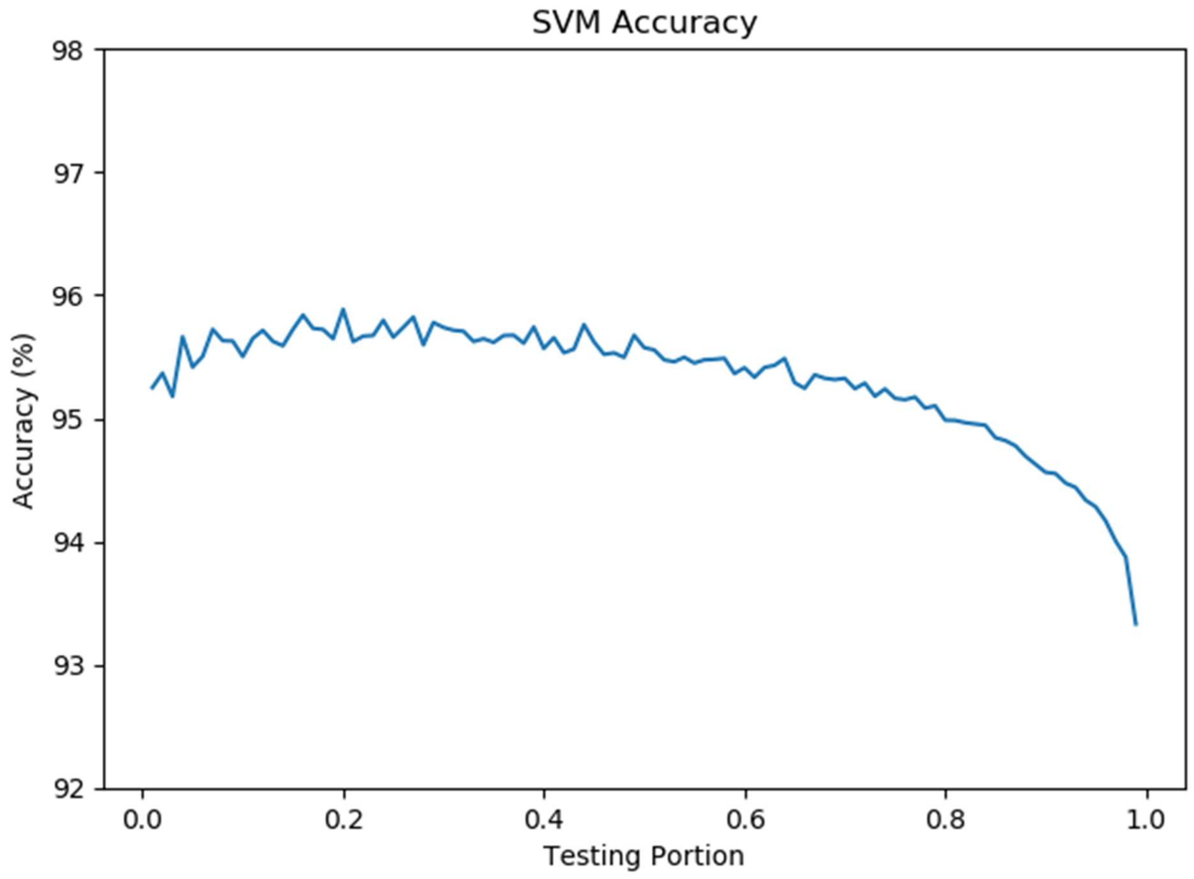

Figure 20. SVM recognition accuracy as a function of dataset test ratio.

The shape of the function is like that of KNN, although, notably, there is a quicker decrease in accuracy beginning in the middle training ratios; accuracy remains high throughout. The shape of the SVM plot is closer to expected, and an accuracy of ~95% is reasonably good without being suspiciously high.

ANNs can automatically learn features from raw data, as they map complex nonlinear functions from the input sensor space to the output class space. ANNs can take a long time to train, although parallelization leveraging GPUs can reduce computation time. The ANN is implemented in keras for python running tensorflow on the backend. The ADAM optimizer was chosen, as it resulted in higher accuracies and faster convergences. Details about the optimizer can be found in [31].

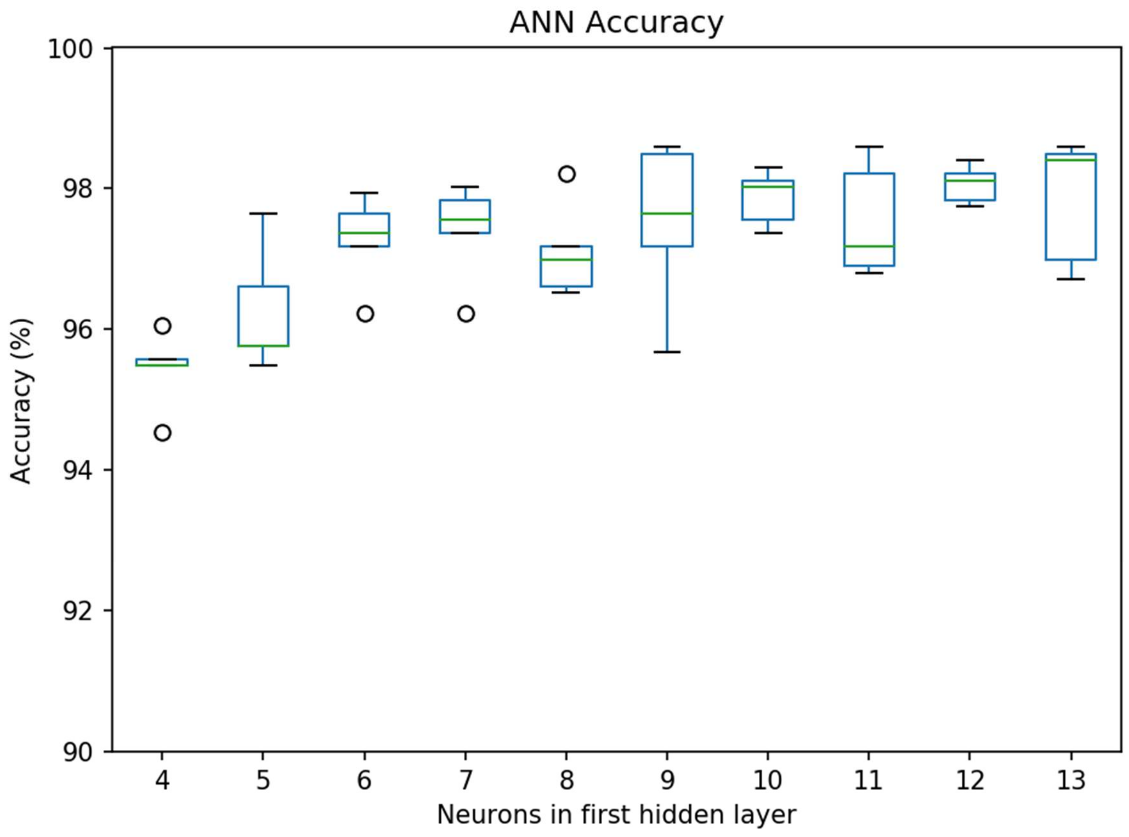

Figure 21. Effect of neuron number in the first hidden layer on recognition accuracy.

Next, the number of neurons in the first hidden layer was varied, and the change in accuracy was observed. Seventy percent of the dataset was used for training, and the remaining 30% was used for testing. The network was trained for 100 epochs with batches of 20 samples. All parameters, other than number of neurons in the first hidden layer, were held constant. Accuracy was calculated five times with various initialized weights, and results were plotted in a box plot. Results are shown above in Figure 21.

Accuracy held roughly constant for the range tested, with a slight increase given additional neurons. Accuracy can be improved by increasing the number of training epochs. Notably, trends in accuracy were more important than accuracy itself at this stage. Additional training epochs will be used when evaluating the final model. Observed variation was likely due to the randomization of connection weights at the start of training.

For the ANN model, 12 neurons were determined to be the optimal choice. According to the plot above, any neurons in the range of six to 13 would be acceptable. Twelve were chosen due to high accuracy with lower variance and also because it was optimal for an object with six degrees of freedom to provide complete information about motion in 12 states. The procedure above was repeated for the second hidden layer, and results are shown below in Figure 22.

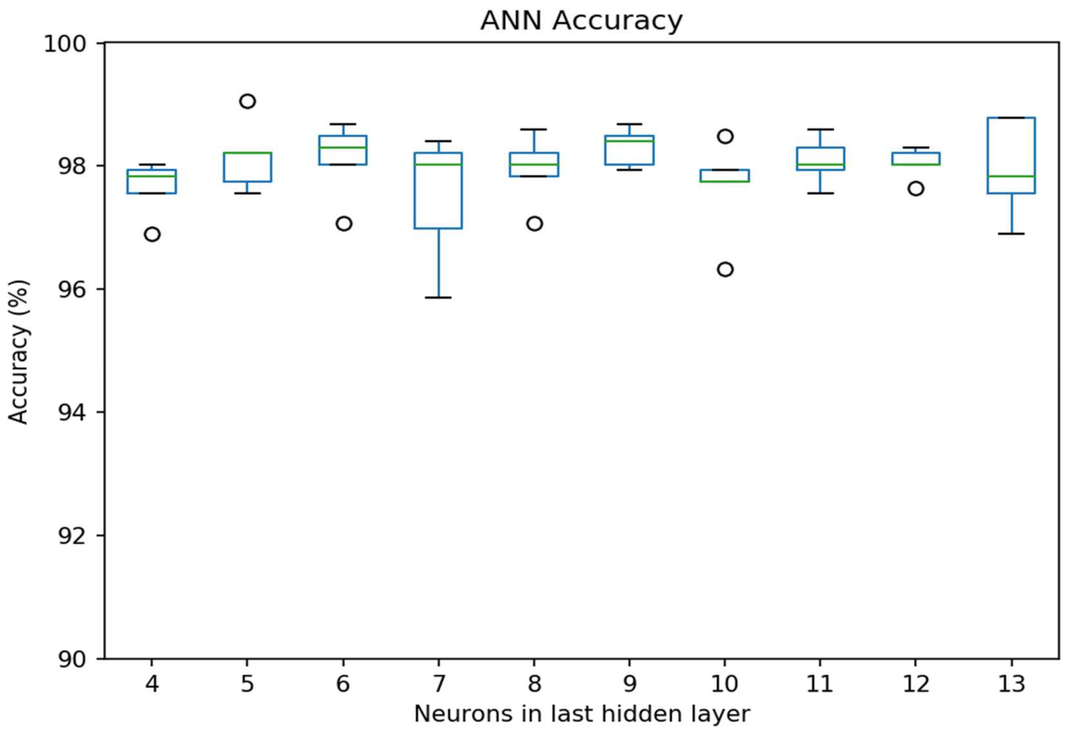

Figure 22. Effect of neuron number in the last hidden layer on recognition accuracy.

In the second hidden layer case, nine neurons were chosen due to the occurrence of highest average; however, significant variation was not observed across the various number of neurons. Overall, the final design includes an input layer with six neurons for the six input data dimensions; a first hidden layer with 12 neurons; a second hidden layer with nine neurons; and an output layer with one neuron. Each layer is fully (i.e., densely) connected to the layers immediately preceding and succeeding. The final ANN diagram is shown below in Figure 23.

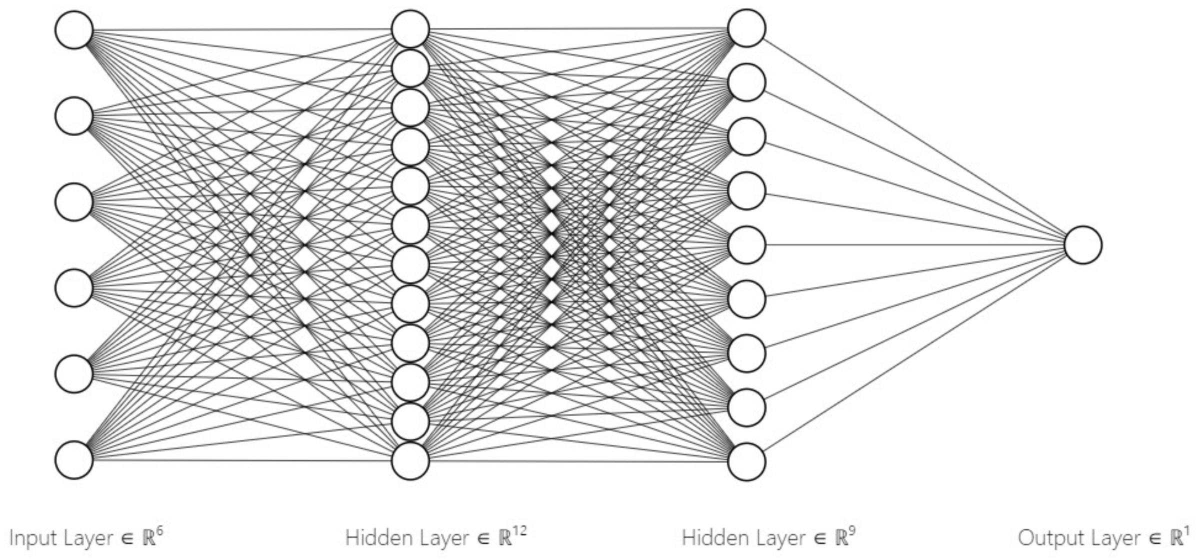
Input Layer $\in \mathbb{R}^6$
Hidden Layer $\in \mathbb{R}^{12}$
Hidden Layer $\in \mathbb{R}^9$
Output Layer $\in \mathbb{R}^1$

Figure 23. Neural network diagram.

The ANN was trained for 300 epochs with a training batch of 20 (i.e., the approximate motion frame size). ANN training history is shown below in Figure 24.

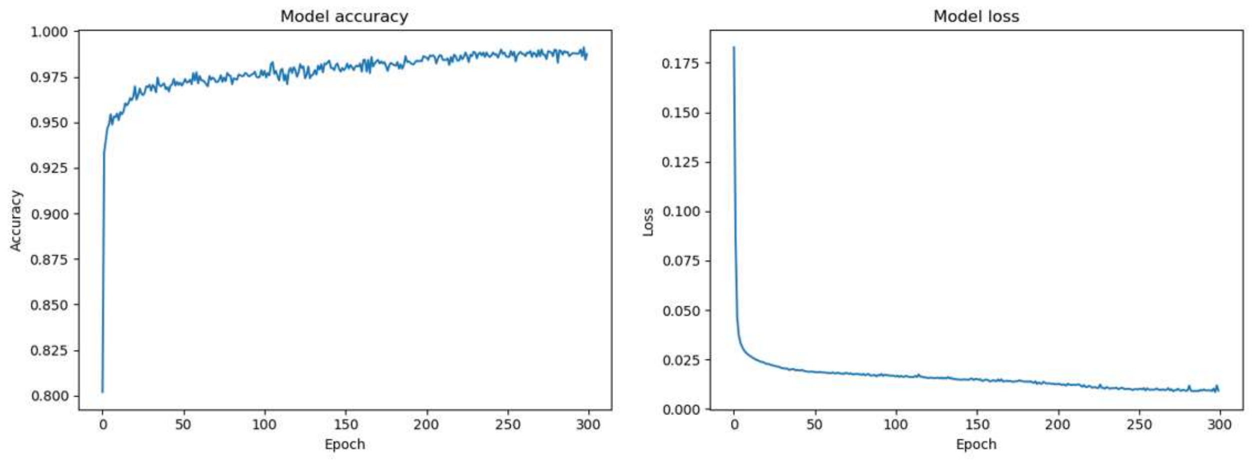

Figure 24. Training history for ANN (See accuracy left and MSE right).

The accuracy (or loss) of the models increases (or decreases) sharply at the beginning of the training and levels out as the training epoch number increases. As with the other models above, the dataset testing ratio was varied, and a subsequent change in accuracy was observed. Results are given below in Figure 25.

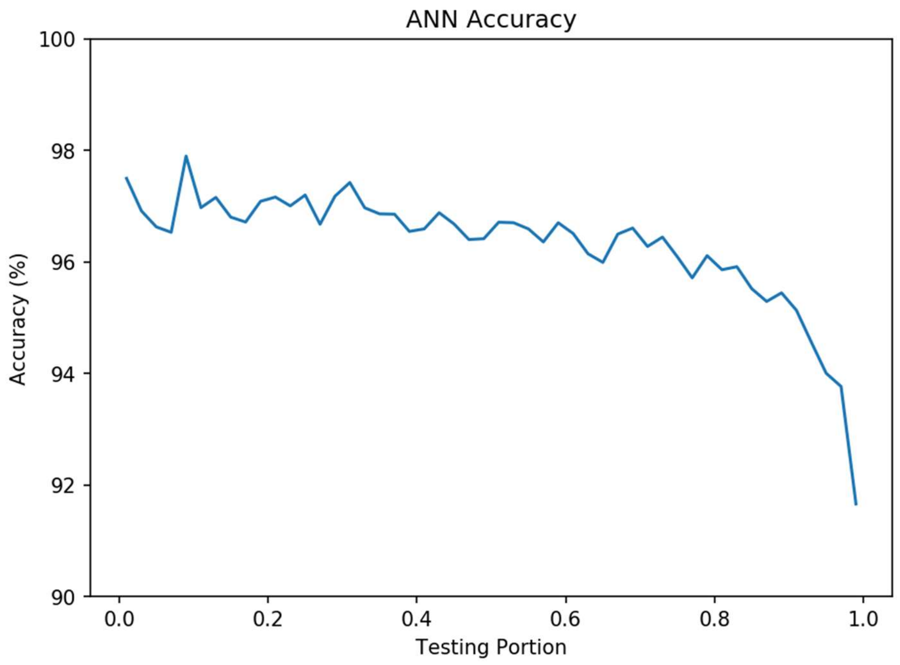

Figure 25: ANN accuracy vs dataset test ratio.

As expected, a gradual decrease in accuracy was observed as the testing portion increased and training portion decreased. ANN was still quite accurate at ~95% even when only 10% of the dataset was used for training.

4. Results and Discussion

A comparison was conducted between KNN, SVM, and ANN model training. The three methods were simultaneously evaluated for various numbers of dataset samples. KNN had k=1; SVM used a Gaussian radial basis kernel function; and ANN was trained for 300 epochs with a training batch size of 20 (i.e., the approximate motion frame size). Included data was the first X sequential data samples in time, with X as the independent variable. Data was included sequentially, not through random sampling. After the initial selection, 70% was used for training and 30% was used for testing. Accuracy for each model was computed five times for each model, and then averaged. Results are shown below in Figure 26.

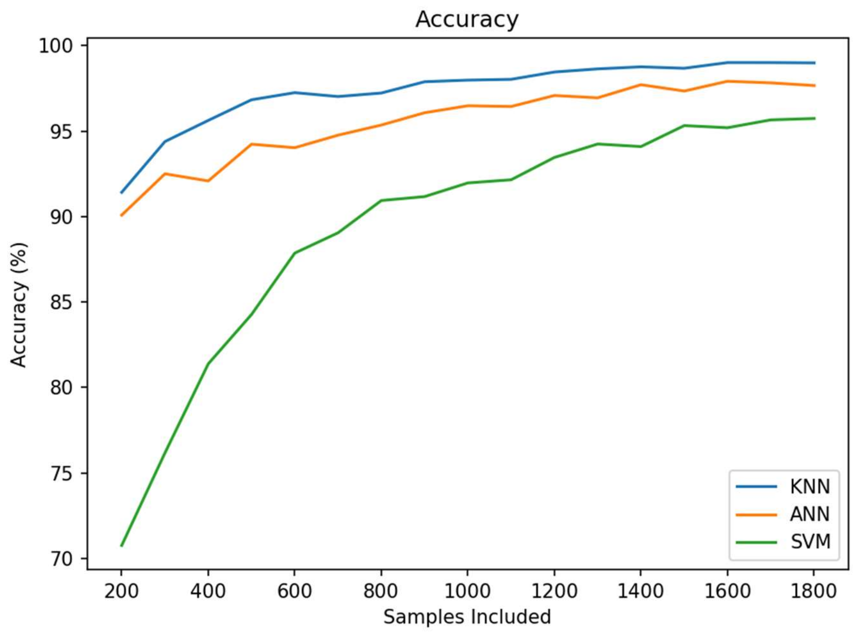

Figure 26. Gesture recognition accuracy vs. samples included for the three models.

Accuracies for each model asymptotically approached a maximum. Only SVM was significantly affected at < 80%. The best accuracy for KNN was 98.95%; the best for SVM was 95.74%; and the best for ANN was 98.12%. Model order remained unchanged across all samples included, with KNN leading, followed by ANN, and SVM last.

All three classifiers distinguished between a smoking gesture and the opening of a pill bottle gesture by way of raw standardized inertial measurements. KNN performed the best and required no real training, although computations were sometimes taxed given a large number of training points. ANN followed closely behind with a longer training time (e.g., 54.4 secs for ANN; 20 ms for KNN) and quicker evaluation time (e.g., 9.9 ms for ANN; 24.9 ms for KNN).

4.1 K-Nearest Neighbor

The final configuration of the KNN classifier used k = 1 with 70% of the dataset used for training and the remaining 30% used for testing. The classifier was executed 1,000 times to mitigate the effects of the random training sampling. Averages for 1,000 test results are presented in Tables 1 and 2 below.

Table 1. Confusion Matrix for KNN Classifier

Confusion Matrix:		Predicted Class		
		Bottle	Smoking	Recall:
True Class	Bottle	527.3	4.0	0.9925
	Smoking	7.3	524.4	0.9863
	Precision:	0.9863	0.9924	

Table 2.: Classification Report for KNN Classifier

Report:				
	precision	recall	f1-score	support
Bottle	0.9863	0.9925	0.9894	531.3
Smoking	0.9924	0.9863	0.9893	531.7
avg	0.9894	0.9894	0.9894	1063
accuracy	0.9895			1063

The KNN classifier performed extremely well when distinguishing between the two gestures. Overall accuracy was 98.95%. Precision for recognizing the bottle was 98.63%, while recall for the bottle was 99.25%. The high recall indicates that the model is very likely to detect when the bottle is actually being opened. The slightly lower precision value (though still quite high) indicated that it is slightly less likely to misclassify the bottle opening gesture for a smoking gesture, rather than vice versa. The averaged confusion matrix demsonstrates this well, as false positives (e.g., bottle opening gesture erroneously detected for smoking gesture) was 7.3 higher than false negative 4.0.

Overall, results should be further verified. The limited dataset suggests that the classifier is overfitted to the two indiviuals and their performed gestures. KNN used k=1, the most simple case (i.e., the model simply assigned unkown points based on which training point was closest). Doing so worked well for this particualar model, as both gestures resided in different domains of the sample space. It is doubtful, however, the model will generalize well given additional individuals and more gestures.

4.2 Support Vector Machine

The final configuration of the SVM classifier utilized a Gaussian radial basis function as its kernel function for increasing dimensionality. Seventy percent of the dataset was used for training with the remaining 30% used for testing. The model was trained 1,000 times, and averages were calculated in order to reduce the effects of the random training sample selection. Results are shown below in Tables 3 and 4.

Table 3. Confusion Matrix for SVM Classifier

Confusion Matrix:		Predicted Class		
		Bottle	Smoking	Recall:
True Class	Bottle	514.5	17.7	0.9667
	Smoking	28.1	502.7	0.9471
	Precision:	0.9482	0.9660	

Table 4. Classification Report for SVM Classifier

Report:				
	precision	recall	f1-score	support
Bottle	0.9482	0.9667	0.9574	532.2
Smoking	0.9660	0.9471	0.9564	530.8
avg	0.9571	0.9569	0.9569	1063
accuracy	0.9574			1063

The SVM classifier performed well when distinguishing between the two gestures. Overall accuracy was 95.74%. Precise recognition the bottle gesture was 94.82%; recall was 96.67%.

High recall indicates that the model satisfactorilly recognized the gesture. A slighly lower precision value (although still quite high) indicated that SVM is more likely to misclassify the smoking gesture as a bottle opening gesture than the other way around. The averaged confusion matrix demsonstrates this phenomenon well, as the false positives (i.e., bottle opening gesture detection as a smoking gesture) at 28.1; false negative were 17.7.

Results proved realistic: ~95% accuracy was measured in other studies that utilized an SVM for classifying human gestures [21, 22]. These results are more believable than the 99% accuracy indicated with KNN. SVM can't overfit as badly as KNN because SVM must consider all training datapoints, while KNN with k=1 only considers the nearest training samples for evaluation.

4.3 Artificial Neural Network

The ANN classifier used an architecture with six input neurons, 12 neurons in the first hidden layer, nine neurons in the second hidden layer, and a single output neuron for estimating the class. All layers were densely connected. The model was trained in 300 epochs using a batch size of 20 data samples (i.e., approximate gesture period). Seventy percent of the dataset was used for training with the remaining 30% used for testing. ANN was trained 10 times, and averages were measured to reduce effects of random weight initialization.

Table 5. Confusion Matrix for ANN Classifier

Confusion Matrix:		Predicted Class		
		Bottle	Smoking	Recall:
True Class	Bottle	526.6	6	0.9887
	Smoking	16.8	513.6	0.9683
	Precision:	0.9691	0.9885	

Table 6. Classification Report for ANN Classifier

Report:				
	precision	recall	f1-score	support
Bottle	0.9691	0.9887	0.9788	533
Smoking	0.9885	0.9683	0.9783	530
avg	0.9788	0.9785	0.9785	1063
accuracy	0.9812			1063

The ANN classifier performed extremely well when classifying both gestures. Overall accuracy was 98.12%. Precision for recognizing the bottle opening gesture was 96.91%, while recall was 98.87%. The slightly higher recall value (although precision was still quite high) indicated that the model was slightly more likely to misclassify a smoking gesture as a bottle opening gesture than the other way around. The averaged confusion matrix demsonstrated this well, as false positives (i.e., smoking gesture detected as a smoking gesture) at 16.8 were higher than the false negatives at 6. Although more false positives is unfortunate, the results demonstrated a small number compared to overall samples.

Results of this investigation are promising. ~98% accuracy was reported in papers that utilized neural networks to classify human gestures [24-26]. To fully verfiy the results, more data is needed for a greater number gestures and additional individuals.

5. Conclusion and Future Work

This thesis compares the ability of three common classifier models, namely KNN, SVM, and ANN, to recognize a bottle opening gesture for improving medication adherence. Recall of the bottle opening proved the most important metric, with precision as a support metric. If recall is higher than precision, more false positives than false negatives are reported. This is important for medication adherence so that a system will not misclassify other gestures as a bottle opening gesture and falsely assume that the patient administered the medication.

Overall KNN accuracy was 98.95%. KNN precision for recognizing the bottle opening gesture was 98.63%, while KNN recall for the bottle opening gesture was 99.25%. Overall SVM accuracy was 95.74%. In this case, precision for recognizing the bottle opening gesture was 94.82%, while SVM recall for the bottle opening gesture was 96.67%. Overall ANN accuracy was 98.12%. ANN precision for recoginzing the bottle opening gesture was 96.91%, while ANN recall for the bottle opening gesture was 98.87%. ANN results continue to show promise [24-26]. Results show that KNN and ANN were more accurate than the SVM, although all models classified the gesture with an accuracy greater than 95%. SVM results matched closely with related recent work: ~95% accuracy is possible with this method.

Results for KNN are the least promising, despite high accuracy. In previous studies, much more data processing was needed for good results, which contradicts these findings. Results for KNN are probably due to overfitting the data, which was only available for two gestures performed by two individuals. More data is needed for a greater number gestures performed by additional individuals to validate that KNN is viable for classifying raw accelerometer and gyroscope data.

ANN results are very promising. ANN and SVM training results in a type of hyperplane. In this research, SVM finds the hyperplane for data processed with a Gaussian radial basis function. ANN experienced the same, though time histories can be accounted for. SVM treats all points as existing simultaneously in the sample space, while ANN can learn time dependecies in the data.

All three models could prove useful for medication adherence. While the KNN model requires more data and testing to realize an improvement in gesture recognition, the model performs extremely well given a two-gesture, two-individual classifation case study. Results are likely to deteroriate in the presence of more gestures or more diverse sampling. SVM alone might not reach a level of commercial accuracy, which is why this method was used to suplement the Bayesian network in [22].

The ANN approach is the most promising, offering the greatest flexibility and ability to learn features without extensive preocessing. More validation would be useful to ensure performance remains high when more gestures and additional individuals are included in the study.

Overall, this thesis demonstrates that all three chosen classification models perform well for the given task. Whether these models will generalize to a more varied population remains to be seen. For medication adherence, more work is needed to translate the classifiers into a system that benefits patients. Even though the methods tested are easily implemented, more work is be needed for real-time IoT medication adherence monitoring by a commercial off-the-shelf (COTS) service.

Future work will focus on testing ANN with additional participants and a greater number of gestures. Eventually, the final aim would be to incorporate machine-learning models into a usable system for medication adherence. The best way to reach as many patients as possible is developing a user-friendly, smart watch application for continuously monitoring patients administer medication. Utlizing cloud-based data processing could ease the computational burden on devices. More data processing is needed to slice live data streams from the smart watch into manageable windows for classification. Many researchers are currently investigating these solutions. Thus, a user-friendly, robust medication adherence system should be available in the near future.

References

[1] S. Reif, B. Ayadi, S. Ayadi, N. Pierce, C. Attey, H. Refai, "Human Movement Classifier via k-NN algorithm detecting for opening bottle motion", University of Oklahoma ECE Capstone Project, Spring 2017.

[2] A. Meyer, I. Grady, G. Robinson, A. Horton, P. Velesko, H. Refai, "Movement Classification Based on Time-Series Accelerometer and Gyroscope Data", University of Oklahoma ECE Capstone Project, Spring 2017

[3] NEHI, "Improving patient medication adherence: a $290 billion opportunity," http://www.nehi.net/bendthecurve/sup/documents/Medication_Adherence_Brief.pdf

[4] Marie T. Brown, J. K. (2011). "Medication Adherence: WHO Cares?". Mayo Clinic Proceedings, 304-314.

[5] Jun Qi, Po Yang, Geyong Min, Oliver Amft, Feng Dong, Lida Xu, "Advanced internet of things for personalised healthcare systems: A survey", Pervasive and Mobile Computing, Volume 41, 2017, pp 132-149, https://doi.org/10.1016/j.pmcj.2017.06.018.

[6] As. Agarawala, S. Greenberg, and G. Ho, "The context-aware pill bottle and medication monitor," Technical Report, Department of Computer Science, University of Calgary, Calgary, Canada, 2004.

[7] J. Pak and K. Park, "Construction of a smart medication dispenser with high degree of scalability and remote manageability," *Journal of Biomedicine and Biotechnology*, vol. 2012, 2012.

[8] H. H. Huynh, J. Meunier, J. Sequeira, and M. Daniel, "Real time detection, tracking and recognition of medication intake," *World Academy of Science, Engineering and Technology*, vol. 60, pp. 280–287, December 2009.

[9] G. Bilodeau and S. Ammouri, "Monitoring of medication intake using a camera system," *Journal of Medical Systems*, vol. 35, no. 3, pp. 377–389, June 2011.

[10] F. Hasanuzzaman, X. Yang, Y. Tian, Q. Liu, and E. Capezuti, "Monitoring activity of taking medicine by incorporating RFID and video analysis," *Network Modeling Analysis in Health Informatics and Bioinformatics*, vol. 2, no. 2, pp. 61–70, July 2013.

[11] J. Lundell, T. L. Hayes, S. Vurgun, U. Ozertem, J. Kimel, J. Kaye, F. Guilak, and M. Pavel, "Continuous activity monitoring and intelligent contextual prompting to improve medication adherence," in *Proceedings of 29th IEEE Annual International Conference on Engineering in Medicine and Biology.*

[12] C. Chen, N. Kehtarnavaz and R. Jafari, "A medication adherence monitoring system for pill bottles based on a wearable inertial sensor," 2014 36th Annual International Conference of the IEEE Engineering in Medicine and Biology Society, Chicago, IL, 2014, pp. 4983-4986.

[13] J. Han, M. Kamber, J. Pei, "Classification: Advanced Methods," *Data Mining: Concepts and Techniques,* 3rd ed., Morgan Kaufmann Publishers Inc., San Francisco, 2005, pp. 393-442.

[14] F. G. Hofmann P. Heyer and G. Hommel "Velocity profile based recognition of dynamic gestures with discrete Hidden Markov Models " in Gesture and Sign Language in Human-Computer Interaction vol. 1371 of Lecture Notes in Computer Science pp. 81-95. 1998.

[15] S. Kallio J. Kela and J. Mantyjarvi "Online gesture recognition system for mobile interaction " in Systems Man and Cybernetics 2003 vol. 3 pp. 2070-2076 vol.3.

[16] J. Kela P. Korpipaa J. Mantyjarvi S. Kallio G. Savino L. Jozzo and D. Marca "Accelerometer-based gesture control for a design environment " Personal and Ubiquitous Computing vol. 10 no. 5 pp. 285-299 July 2006.

[17] T. Pylvanainen "Accelerometer Based Gesture Recognition Using Continuous HMMs Pattern Recognition and Image Analysis " vol. 3522 of Lecture Notes in Computer Science chapter 77 pp. 413-430. Berlin Heidelberg 2005.

[18] D. H. Wilson and A. Wilson "Gesture recognition using the xwand " Tech. Rep. CMU-RI-TR-04-57 Robotics Institute April 2004.

[19] J. Liu, Z. Wang, L. Zhong, J. Wickramasuriya and V. Vasudevan, "uWave: Accelerometer-based personalized gesture recognition and its applications," 2009 IEEE International Conference on Pervasive Computing and Communications, Galveston, TX, 2009, pp. 1-9.

[20] A. Akl and S. Valaee "Accelerometer-based gesture recognition via dynamic-time warping affinity propagation & compressive sensing " in Proceedings of the International Conference on Acoustics Speech and Signal Processing 2010.

[21] M. Hoffman P. Varcholik and J. J. LaViola "Breaking the status quo: Improving 3D gesture recognition with spatially convenient input devices " in Virtual Reality Conference (VR) 2010 pp. 59-66.

[22] S.-J. Cho E. Choi W.-C. Bang J. Yang J. Sohn D. Y. Kim Y.-B. Lee and S. Kim "Two-stage recognition of raw acceleration signals for 3-D gesture-understanding cell phones " in 10th International Workshop on Frontiers in Handwriting Recognition 2006.

[23] Jiahui Wu Gang Pan Daqing Zhang Guande Qi and Shijian Li "Gesture recognition with a 3-D accelerometer " 2009 Ubiquitous Intelligence and Computing pp. 25-38.

[24] G. Lefebvre, S. Berlemont, F. Mamalet, and C. Garcia, "BLSTM-RNN based 3D gesture classification " in Proceedings of the International Conference on Artificial Neural Networks 2013 pp. 381-388.

[25] S. Duffner, S. Berlemont, G. Lefebvre and C. Garcia, "3D gesture classification with convolutional neural networks," *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Florence, 2014, pp. 5432-5436.

[26] S. Ha and S. Choi, "Convolutional neural networks for human activity recognition using multiple accelerometer and gyroscope sensors," 2016 International Joint Conference on Neural Networks (IJCNN), Vancouver, BC, 2016, pp. 381-388.

[27] Casey A. Cole, Bethany Janos, Dien Anshari, James F. Thrasher, Scott Strayer, Homayoun Valafar, "Recognition of Smoking Gesture Using Smart Watch Technology", Proceedings of the International Conference on Health Informatics and Medical Systems (HIMS), July 2016, Las Vegas, NV

[28] A. Parate , M. Chiu , C. Chadowitz , D. Ganesan , E. Kalogerakis, "RisQ: recognizing smoking gestures with inertial sensors on a wristband", Proceedings of the 12th annual international conference on Mobile systems, applications, and services, June 16-19, 2014, Bretton Woods, New Hampshire, USA [doi>10.1145/2594368.2594379]

[29] Hong F, You S, Wei M, Zhang Y, Guo Z. MGRA: Motion Gesture Recognition via Accelerometer. Sensors (Basel). 2016;16(4):530. Published 2016 Apr 13. doi:10.3390/s16040530

[30] M. Luk and M. Luk, "Time-Series Analysis: Wearable Devices using DTW and kNN", SFL Scientific – Data Science Consulting & Advanced Analytics, 2017. [Online]. Available: https://sflscientific.com/case-studies/2016/6/4/time-series-analysis-fitbit-using-dtw-and-knn.

[31] D. P. Kingma and J. L. Ba. "Adam: A method for stochastic optimization." 2014. arXiv:1412.6980v9

[32] Pierluigi Casale, O. P. (2011). Human Activity Recognition from Accelerometer Data Using a Wearable Device. Conference: Pattern Recognition and Image Analysis. Las Palmas de Gran Canaria, Spain.

[33] Bruno, B., Mastrogiovanni, F., Sgorbissa, A., Vernazza, T., Zaccaria, R. Analysis of human behavior recognition algorithms based on acceleration data. In: IEEE Int Conf on Robotics and Automation (ICRA), pp. 1602--1607 (2013)