Final Report

James Abundis (jlabundi@calpoly.edu)
Jesse Bao (jlbao@calpoly.edu)
Natalie Keelan (nkeelan@calpoly.edu)

1 Introduction and Previous Work

The primary motivation behind our project is to determine if performing certain activities has an effect on long term health. Unfortunately, it's difficult to obtain data that accurately represents an individual's activity over time. While it may be easy to collect accelerometer data over any given period of time, the same does not hold true for collecting the ground truth—the activity. As a result, the key goal of our work is to create a model that is capable of classifying activities for unlabeled accelerometer data.

This objective has been explored in previous work, as well. Research by Staudenmayer et al., was conducted to estimate the metabolic equivalent of tasks (METs) and classify activity intensity, sedentary time, and locomotion time for lab-created wrist-worn accelerometer data [2]. Another group, Willetts et al., looked to classify six broad activities using free-living wrist accelerometer data from the UK Biobank dataset [3]. One common weakness among these two previous approaches is that they classified fixed-length intervals of time—15s and 30s respectively. This may be problematic, as activities in real world setting are not guaranteed to last the same amount of time, so fixed-length intervals may capture more than one activity, or only a small portion of a longer activity.

Our approach is to first classify the data on a second-by-second basis using neural networks. This allows us to establish a baseline accuracy for our data that we can compare against different methods. One method by Lyden et al., resolves the aforementioned problem by using a technique called the Sojourn Method [1]. This technique is divided into three main tasks: segmentation, classification, and estimation. First, it segments the data by identifying points of rapid acceleration of deceleration. Then, it classifies each bout as either periods of activity or inactivity. Finally, it estimates the MET values of each bout. For the purposes of our project, our interest lies within the segmentation and classification portions of this method. Our goal is to segment our data similarly to the Sojourn Method, and classify each resulting bout with their respective activities.

2 Dataset

Our dataset was collected in a study conducted by Dr. Sarah Keadle. It consists of accelerometer data from twenty-six subjects that each participated in two, two-hour direct

observation periods during which they were video-recorded to establish the ground truth. During each direct observation session, subjects were asked to perform a single type of behavior (Appendix: Table 24) as they would in their everyday lives.

The study utilized two types of activity monitors: BioStamp and Actigraph. The BioStamp monitors were affixed to the medial chest and left anterior thigh, while the Actigraph monitors were worn on the wrist and hip. Each type of monitor collected data at a different frequency. The BioStamp monitors collected data at a rate of 31.25 Hz, while the Actigraph monitors collected data at a rate of 80 Hz. Each monitor provided raw acceleration data for three axes (x, y, and z), and the footage from the direct observation periods was reviewed to manually encode a second-by-second ground truth based on the behavior and posture of the subject (Appendix: Tables 24 and 25).

3 Feature Engineering

Our data consisted of csv files containing a timestamp and raw acceleration values for three axes with varying levels of resolution that had to be merged in with the second-by-second encodings. So, we aggregated the accelerometer data into second-by-second level features. The complete list of features and their descriptions can be seen in the Appendix: Table 23.

Among the features were standard summary statistics for each axis such as the minimum, maximum, mean, and standard deviation. There are also features that are aggregates of the vector magnitude, and accelerations relative to the vertical axis. We also added frequency-level features which were computed via fast Fourier transform.

3.1 Principal Component Analysis

To determine whether all of our features are significant, and to explore the possibility of dimensionality reduction, we performed a principal component analysis on our dataset.

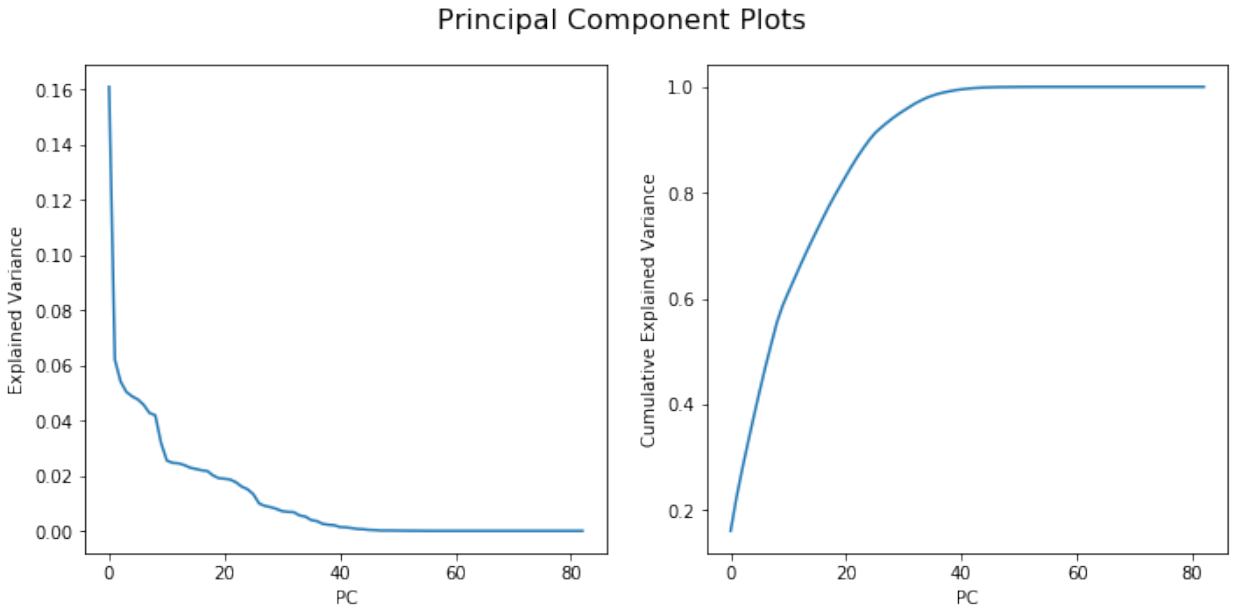

Figure 1: Principal Component Plots

As seen in Figure 1, the maximum percentage of explained variance that any principal component has is about 16%. Additionally, it takes approximately 30 principal components to explain a cumulative 95% of variance.

Principal Component 1		**Principal Component 2**		**Principal Component 3**	
std.vm	0.2647	mean.ang	0.3995	xfft12	0.2594
std.z	0.2639	mean.x	0.3525	xfft5	0.2594
std.x	0.2637	xfft1	0.3498	mfft12	0.2389
std.y	0.2631	mean.z	0.3340	mfft5	0.2389
mean.vm	0.2607	zfft1	0.3323	xfft13	0.2188
pct75.vm	0.2606	max.x	0.3067	xfft4	0.2188
mfft1	0.2536	mean.y	0.1826	zfft12	0.2060
min.x	0.2468	yfft1	0.1794	zfft5	0.2060
pct50.vm	0.2409	pct25.vm	0.1757	xfft14	0.1761
min.z	0.2398	mean.yz	0.1654	xfft3	0.1761

Table 1: Top 10 coeffecients for Principal Components 1, 2, and 3

If we look at the top coeffecients for each principal component, we can see a general trend for what they represent. In Table 1, which shows the top 10 coeffecients for principal components 1, 2, and 3, we can see that principal component 1 consists mostly of vector magnitude features and summary statistics for single axes. Principal component 2 is similar

to the first, except it has the inclusion of the mean angle relative to the vertical. Principal component 3 and onward are mainly combinations of the various fast Fourier transform features. This indicates that the reduction in dimensionality is achieved mainly by selecting the more significant fast Fourier transform features.

4 Broad Activity Encoding

Because previous work suggested that these categories may be too granular to reasonably classify, we created a broader ground truth encoding called Broad Activity. The Broad Activity category was encoded to match the activity encoding from a previous paper by Matthew Willets. However, Willets paper included sleep as one of the activity encodings, while none of our includes sleeping. Consequently, we encoded our data to the other 5 categories in Willets paper using our original ground truth encodings: Coding, Activity Per Second, and Posture Per Second. The 5 categories are as follows: Sit/Stand, Walking, Mixed Activity, Vehicle, Bicycling. Table 1 shows the descriptions for each category, as well as the number of seconds of data we have for each. As shown in Table 1, we have far more Sit/Stand data than any other category. We also provided 4 graphs in Appendix C to show how the accelerometer data differs by Broad Activity category and by monitor.

Broad Activity	**Description**	**Counts**
Sit/Stand	Sitting, lying, or standing still	246,145
Walking	Walking, running, or walking up or down stairs	56,937
Mixed Activity	Stand and move	21,842
Vehicle	Sitting in a vehicle	13,656
Bicycling	Bicycling	6,108

Table 2: Broad Activity Encoding with Counts

4.1 Neural Network Results

Table 3: Single Sensor: Thigh Broad Activity

		Predicted					
		Bicycling	Mixed-Activity	Sit/Stand	Vehicle	Walking	Recall
Actual	Bicycling	4213	494	381	13	1007	**0.69**
	Mixed-Activity	58	7010	15569	27	5753	**0.247**
	Sit/Stand	142	5282	237413	11745	2007	**0.925**
	Vehicle	7	22	7049	6555	23	**0.48**
	Walking	133	3932	2618	16	50520	**0.883**
	Precision	**0.925**	**0.419**	**0.903**	**0.357**	**0.852**	**0.845**

4.2 Segmentation and Neural Network Results

5 Detailed Activity Encoding

Although our best model with the thigh monitor was classifying Broad Activity with 84.5% overall accuracy, we were not completely satisfied with those results. We were able to classify Bicycling, Sit/Stand, and Walking with relatively high precision, however we classified Mixed-Activity and Vehicle with fairly low precision. Thus, we determined that the Broad Activity Encoding might not be the best activity encoding for classifying activities using our accelerometer data. Consequently, we created a new activity encoding in hopes of increasing our predictive power.

This second activity encoding, called the Detailed Activity Encoding, contains 7 activities: Vehicle, Sit/Stand, Stand and Move Light, Stand and Move Moderate or Vigorous, Walking, Running, and Bicycling. The only two activities that remain consistent from the Broad Activity Encoding to the Detailed Activity Encoding are Bicycling and Vehicle. The descriptions and counts (in number of seconds) for all 7 activities can be found in Table 4 below.

For the Broad Activity Encoding, Mixed-Activity was often misclassified as either Sit/Stand or Walking. It is intuitive that Mixed-Activity, stand and move, was often misclassified as Sit/Stand because the Sit/Stand category includes standing still. Thus, we chose lump standing and moving lightly and standing still into together into a category called Stand and Move Light in our Detailed Activity Encoding. Standing still cannot fall under Stand and Move Moderate or Vigorous because standing still is not considered a "moderate" or "intense" movement. Also, by removing standing still from Sit/Stand, we ended up with a Sit/Lie category in our Detailed Activity. Sit/Lie only pertains to sitting or lying as the name suggests.

Similarly, it is intuitive that Mixed-Activity was often misclassified as Walking. In the Broad Activity Encoding, the only discrepancy between Mixed-Activity (stand and move) and Walking is the number of steps a person takes. If a person takes less than 10 steps in a row, then that activity is considered stand and move. However if a person were to take 10 or more steps in a row, that activity is considered walking. To emphasize a discrepancy between walking and stand and move, we chose to consider walking consecutively for one minute or longer to be categorized as Walking in our Detailed Activity Encoding. On the other hand, walking consecutively for less than one minute would be lumped together with stand and move into one of the Stand and Move categories in our Detailed Activity Encoding. Depending on the intensity of the movement, the activity could either be considered Stand and Move Light or Stand and Move Moderate or Vigorous.

Although we were not classifying walking with poor precision, we still wanted to split running and walking into two different categories in our Detailed Activity Encoding. Therefore, we ended up with a Running category in our Detailed Activity Encoding that contains only running. We also ended up with a Walking category in our Detailed Activity Encoding that contains walking consecutively for one minute or longer.

Detailed Activity	**Description**	**Counts**
Sit/Lie	Sitting or lying	166,720
Stand and Move Light	Stand Still, Stand and Move, or Walk for Less than 1 minute Lightly	121,936
Walking	Walking for longer than 1 minute	33,796
Vehicle	Sitting in a vehicle	13,656
Stand and Move Moderate or Vigorous	Stand and Move or Walk for Less than 1 minute Moderately or Vigorously	10,524
Running	Running	9,249
Bicycling	Bicycling	6,108

Table 4: Detailed Activity Encoding with Counts

The results for our best Neural Network model for Detailed Activity are shown in below Table 5 below. The best model again used the Thigh monitor data. The overall accuracy of 86.9% in this model shows improvement from the previous accuracy of 84.5%. However, we were still unsatisfied with these results, especially regarding the Stand and Move Moderate or Vigorous and Vehicle categories. Therefore, we decided to make another activity encoding called Final Activity Encoding.

Table 5: Single Sensor: Thigh Detailed Activity

		Predicted							
		Bicy-cling	Run-ning	Sit/Lie	Stand & Move Light	Stand & Move Moderate or Vigorous	Vehicle	Walking	**Recall**
Actual	Bicylcing	3727	52	1202	299	310	1	517	**0.61**
	Running	13	8661	13	344	14	1	203	**0.936**
	Sit/Lie	263	4	154773	2143	4	9344	189	**0.928**
	Stand & Move Light	208	252	874	115015	47	71	5469	**0.943**
	Stand & Move Moderate or Vigorous	81	242	70	5949	48	13	4121	**0.005**
	Vehicle	14	0	8409	28	1	5179	25	**0.379**
	Walking	59	188	5	6309	151	0	27084	**0.801**
	Precision	**0.854**	**0.921**	**0.936**	**0.884**	**0.083**	**0.355**	**0.72**	**0.869**

6 Final Activity Encoding

The Final Activity Encoding made two very minor changes from the Detailed Activity Encoding. First, we determined that Stand and Move Moderate or Vigorous had too little data and was being classified unsuccessfully with a precision of 8.3% and recall of 0.5% (as seen in Table 5). The vast majority of the misclassifications were being classified at Stand and Move Light. Thus, we decided to lump the Stand and Move Light and Stand an Move Moderate or Vigorous into one category called Stand and Move in our Final Activity Encoding.

Secondly, we determined that Vehicle was often being misclassified as Sit/Lie and Sit/Lie was relatively often being misclassified as Vehicle. This intuitively makes sense because the description for Vehicle is: sitting in a vehicle. Therefore, we chose to lump vehicle in with the sit/lie category, and again call it Sit/Lie in our Final Activity Encoding.

Lastly, the other 3 categories: Walking, Running, and Bicycling remained the same. Thus, our Final Activity Encoding has 5 categories: Sit/Lie, Stand and Move, Walking, Running, and Bicycling. The descriptions and counts (in seconds) for the Final Activity Encoding can

be found in Table 6.

Final Activity	Description	Counts
Sit/Lie	Sitting or lying (including sitting in a vehicle)	180,376
Stand and Move	Stand Still, Stand and Move, or Walk for Less than 1 minute	132,460
Walking	Walking for longer than 1 minute	33,796
Running	Running	9,249
Bicycling	Bicycling	6,108

Table 6: Broad Activity Encoding with Counts

Figure 2 below shows an example of how mean vector magnitude differs per Detailed Activity over about 10,000 seconds for a single participant. As we would expect, Running has a periodic thigh movement with the highest intensity. Walking also has a periodic thigh movement with the next highest intensity of the Final Activities. Again intuitively, Sit/Lie has the smallest amount of thigh movement. Also, stand and move seems to be a bit more sporadic and unpredictable than the other activities as we would expect. Lastly, bicycling seems to have periodic thigh movement with lower intensity than we see with running or walking.

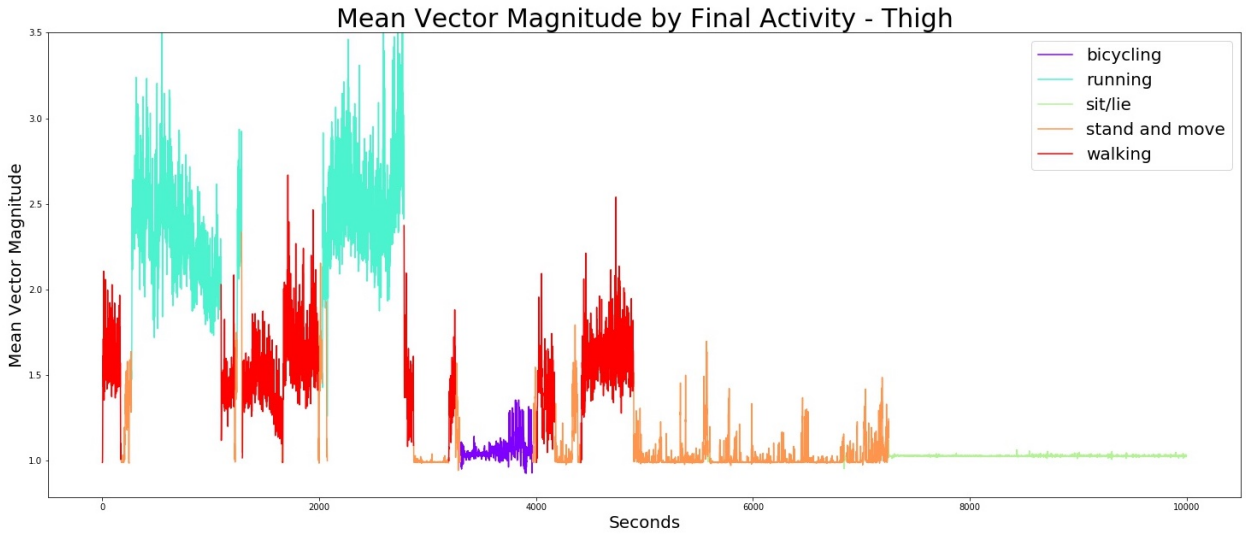

Figure 2: Mean Vector Magnitude by Time, Colored by Final Activity - Thigh

7 Model Exploration

In the next phase of our capstone we sought to build as many models as possible for classifying our best encoding: final activity. The models built and tested include: neural networks,

random forests, decision trees, and neural networks with segmentation. Along with building these models our team performed minor parameter tuning and applied model boosting to some models to improve model performance. We understand that other models exist for classification, but given the time constraint of this project we were only able to produce the above.

7.1 Single Sensor Neural Network

For our first model we used a simple feed forward, fully connected neural network taking second by second input for the 83 features. This neural network is designed with a funnel down architecture as seen in Figure 2. The justification for using an architecture that funnels down is that our original features are not all important in classifying final activity. This claim about our features is supported in our PCA analysis that showed the 83 features can be reduced to 35 principal components that explain 95% of the variance in our final activity. By this analysis, we can say that a funnel down architecture is ideal because each consecutive layer of our neural network will be reducing the dimensionality of the original features to just a few.

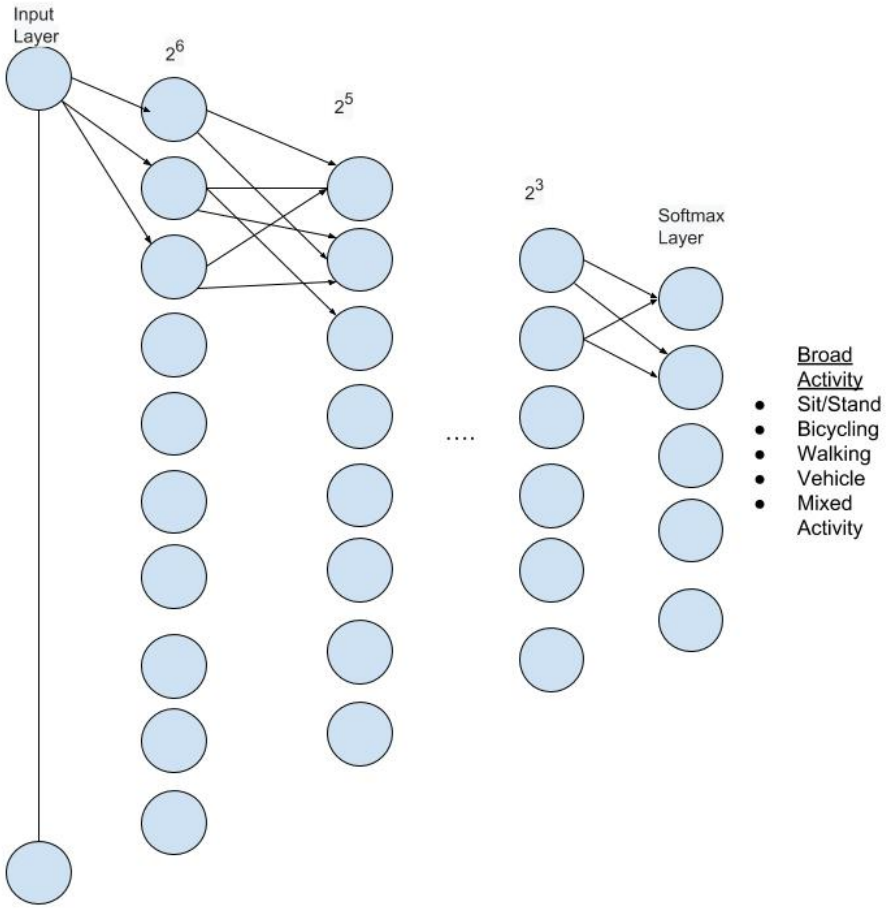

Figure 3: Simple Feed Forward, Neural Network Architecture

For neural network parameter tuning we grid searched the number of epochs to train our neural network with. It is important to do this because increasing the number of epochs, the number of times we train on the training data, can increase model accuracy to some extent. Conversely, if we train with too many epochs we can overfit our neural network on our training data. Generally, there is a maxima for test accuracy versus training accuracy. Training accuracy will continue to increase with more epochs, while test accuracy at some point will begin to decrease as the neural network begins to overfit. Our epoch optimization graph seen in Figure 3 shows that despite increasing the number of epochs the training accuracy does not seem to decrease as a result of overfitting for the epoch range we grid searched. This could be a result of the large dataset size that makes overfitting difficult. However, after about 20 epochs the increase in test accuracy becomes marginal. With this graph we decided that 30 epochs will be used for training our neural networks as it ensures a boost in test accuracy, but isnt too large that our training process takes an unreasonable amount of time to complete. The number of epochs is costly for training considering how large the dataset is, so we made sure to keep our number of epochs reasonable.

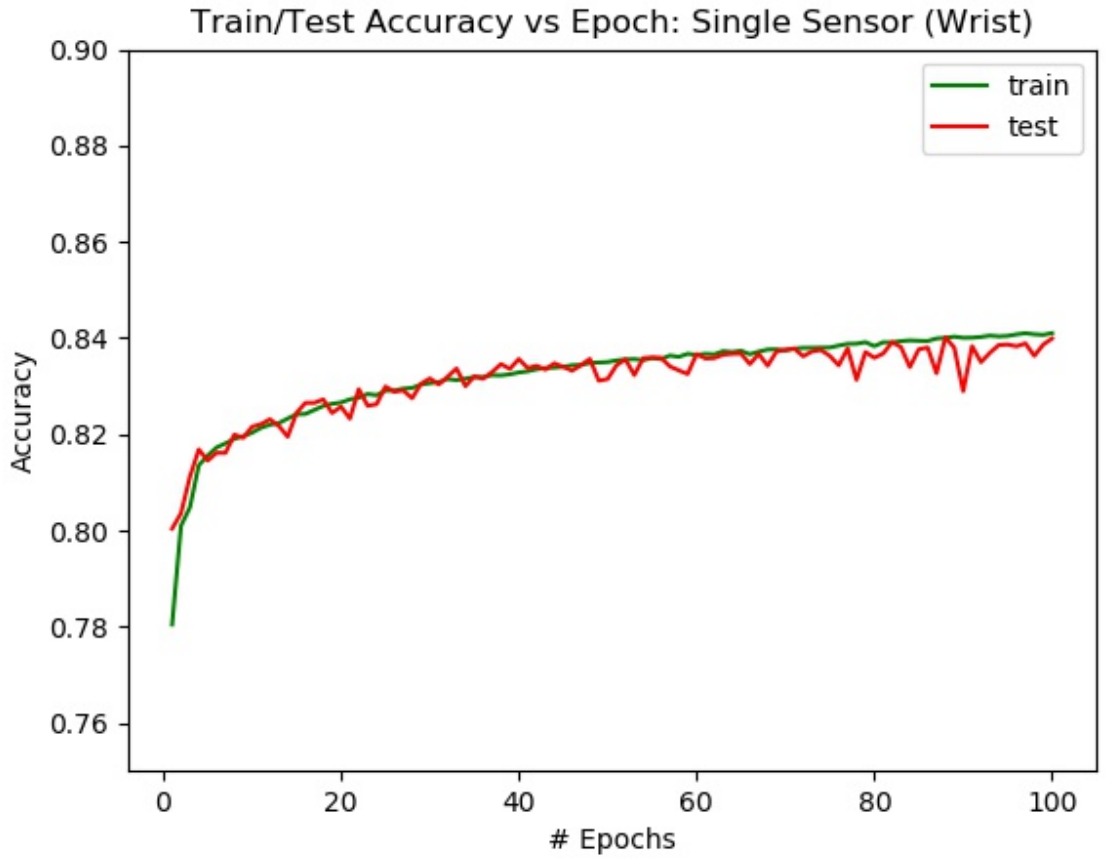

Figure 4: Training/ Test Accuracy vs Number of Epochs

7.2 Single Sensor Neural Network with Boosting

One of the major concerns we had in building a classifier was its ability to account for the class imbalances in the dataset. As stated earlier in the report, we have some final activities such as "sit/lie" that are well represented in our dataset (see Table 5) and other activities such as "running" and "bicycling" that only have a couple thousand data points. In previous work done by Sam Hsu she used Ada-Boosting to address class imbalances and to improve her classifier.

Ada-Boosting in summary takes into account mis-classification errors and uses these errors to reweigh the datapoints that were mis-classified. For example data points that are classified correctly would be given a standard weight of 1 versus data points that are mis-classified which would be given a higher weight. These weights are used to produce a second version of the dataset that with resampled data points based on these weights. Data points with higher weights (mis-classified) will be duplicated in this second dataset, while data points that have lower weights (classified correctly) will have about the same frequency. With this resampling, the second version of our dataset will have a more equal representation of our classes and the model training on this dataset will in theory have better results.

In our custom application and implementation of Ada-Boosting we built an ensemble of models (in this case neural networks). These models are built on datasets that are progressively resampled to better represent our final activity classes. The pipeline for this ensemble can be described by the following:

1. Train model on the training dataset
2. Make predictions on the test dataset and calculate the mis-classifiation errors for each data point
3. Calculate the error of the model for this iteration
4. Use the mis-classification errors to give each data point a new weight
5. Resample the data with new weights to create a new training dataset
6. Give this new training dataset to the next iteration
7. Go back to step 1 and repeat this loop for n estimators

Finally, in order for the completed ensemble to make a prediction it uses a "majority rule" vote using all of the models' predictions in the ensemble. It is important to note that this "majority rule" takes the *weighted* predictions of each model in the ensemble. The weight given to each model is decided by the model error. Assigning model weights in this way gives models with more accurate predictions more say in this "majority rule" vote versus weaker, less accurate models.

This meta-algorithm described above is considered a supervised machine learning method because it is up to us how many estimators (models) we use for creating this ensemble. Ideally this leaves it up to our team to grid search the optimal number of estimators for the

ensemble for increasing prediction accuracy. The results for an ensemble with Ada-Boosting are shown in Figure 4.

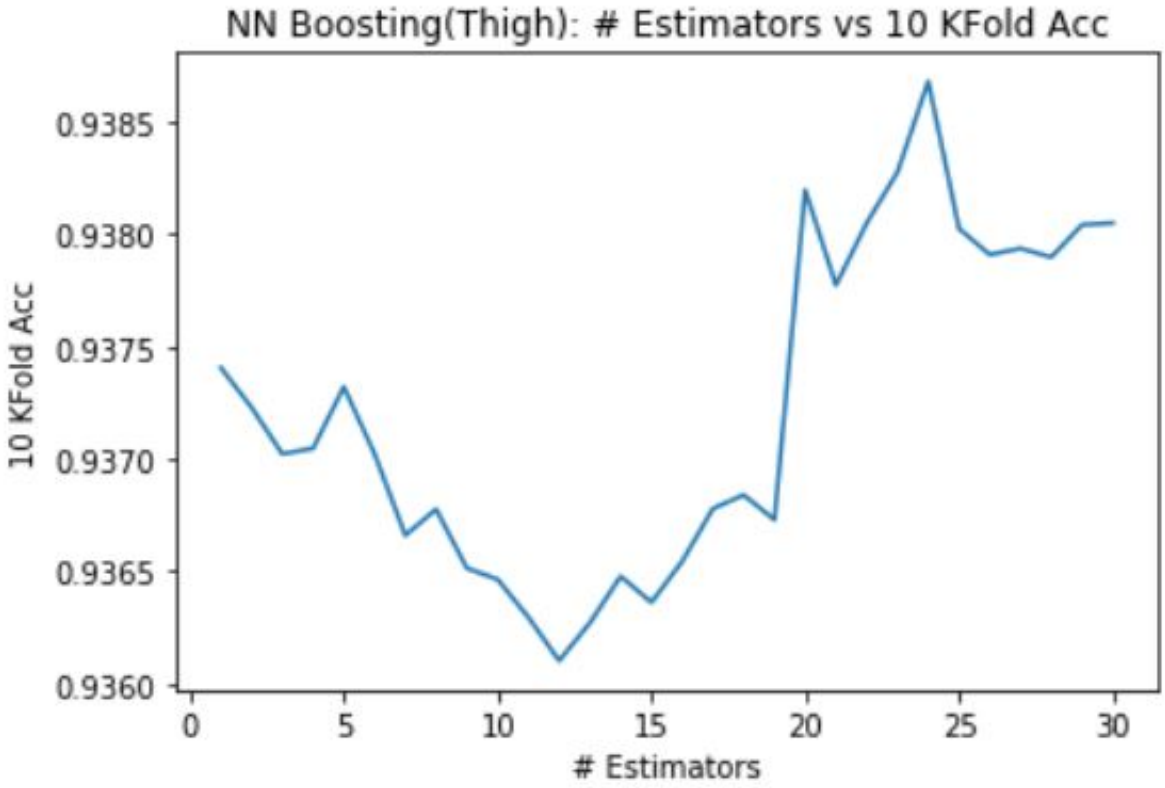

Figure 5: Neural Network Boosting

In our grid search for the number of estimators, we used the neural network for the thigh sensor because our preliminary results so far shown that the thigh sensor had the highest accuracy overall. From here we wanted to see if it was possible to improve this neural network through Ada-Boosting. As seen in Figure 4, Ada-Boosting is largely ineffective in improving prediction accuracy for the number of estimators we grid searched. Our reasoning for the lack of improvement is that our neural network for the thigh sensor already has a high prediction accuracy of 93.9% (see Table 7). In our background research for Ada-Boosting, it is stated that Ada-Boosting works better in improving a weak classifier(s) [4]. Our baseline model starts at such a high accuracy even with class imbalances in our dataset that it may be making the applied Ada-Boosting ineffective.

7.3 Two Sensor Neural Network

On top of building single sensor neural networks the team wanted to explore whether combining sensors would improve classification accuracy. The theory behind this is that two

sensors together may better identify some physical activities that we are trying to classify than a single sensor. To make a two sensor neural network we kept the same funnel down neural network architecture, but increased the shape of our input. Instead of 83 features, we would have 83 features from our first sensor and 83 features from our second sensor to make a total of 166 features. To accommodate the increased input dimension we added additional layers to the neural network architecture.

7.4 Random Forest Classifier

The next model we used for classification is a random forest. For this classifier we did some minor parameter tuning such as setting the maximum depth of the random forest. Other parameters for the random forest were left as the defaults provided by the Sklearn library. In summary, tuning this parameter controls the the amount of information the random forest holds from the data. The deeper the random forest, the more splits it has for capturing information. The results for the random forest depth parameter tuning are shown in Figure 5.

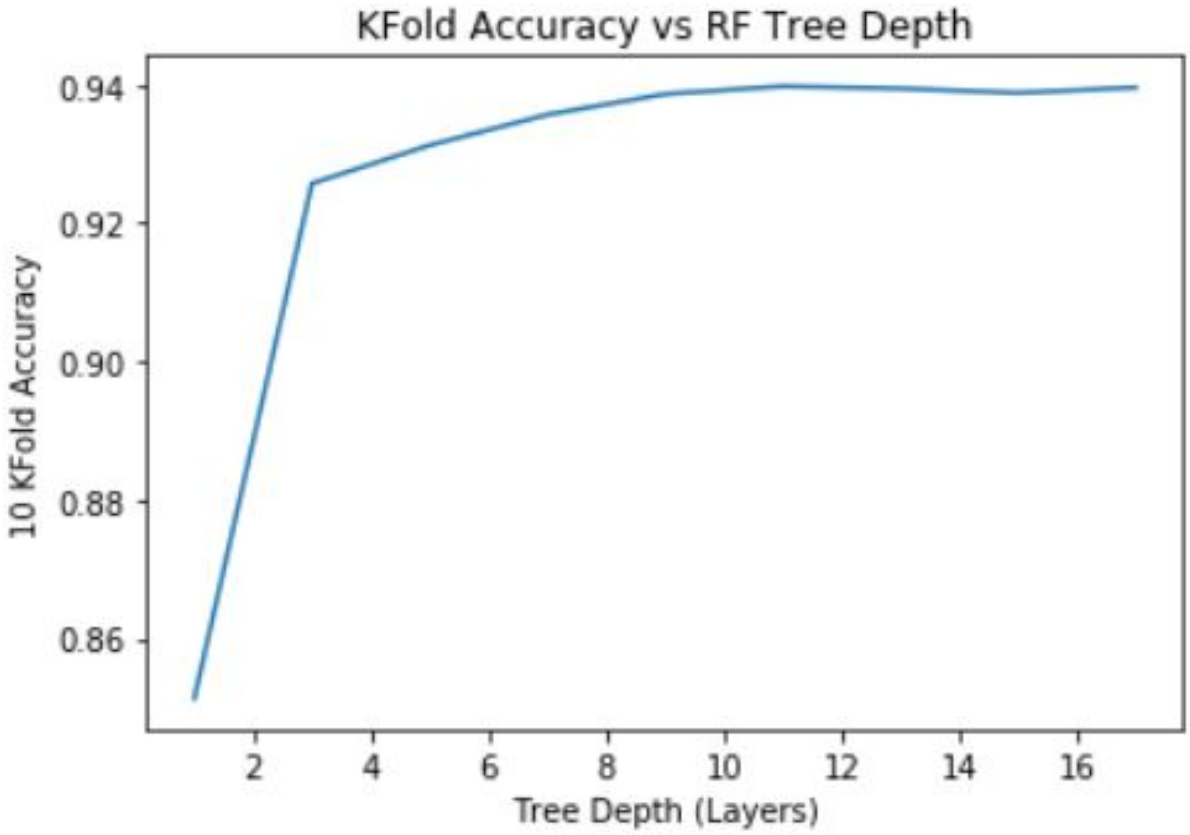

Figure 6: Parameter Tuning: Tree Depth

As seen above, we have a notable elbow in our accuracy graph at a tree depth of 4. However,

even after this elbow there are still gains in accuracy for increasing tree depth. We decided to use a tree depth of 9 looking at the above graph. Increasing tree depth after that shows little to no increase in prediction accuracy.

7.5 Random Forest Classifier with Boosting

To improve our random forest model we also applied Ada-Boosting. Instead of using our own implementation of Ada-Boosting, which was customized for a neural network, we used Sklearn's AdaBoostClassifier which applies Ada-Boosting to an ensemble of models (in this case random forests). Our results for an ensemble of Ada-Boosted random forests are shown in Figure 6.

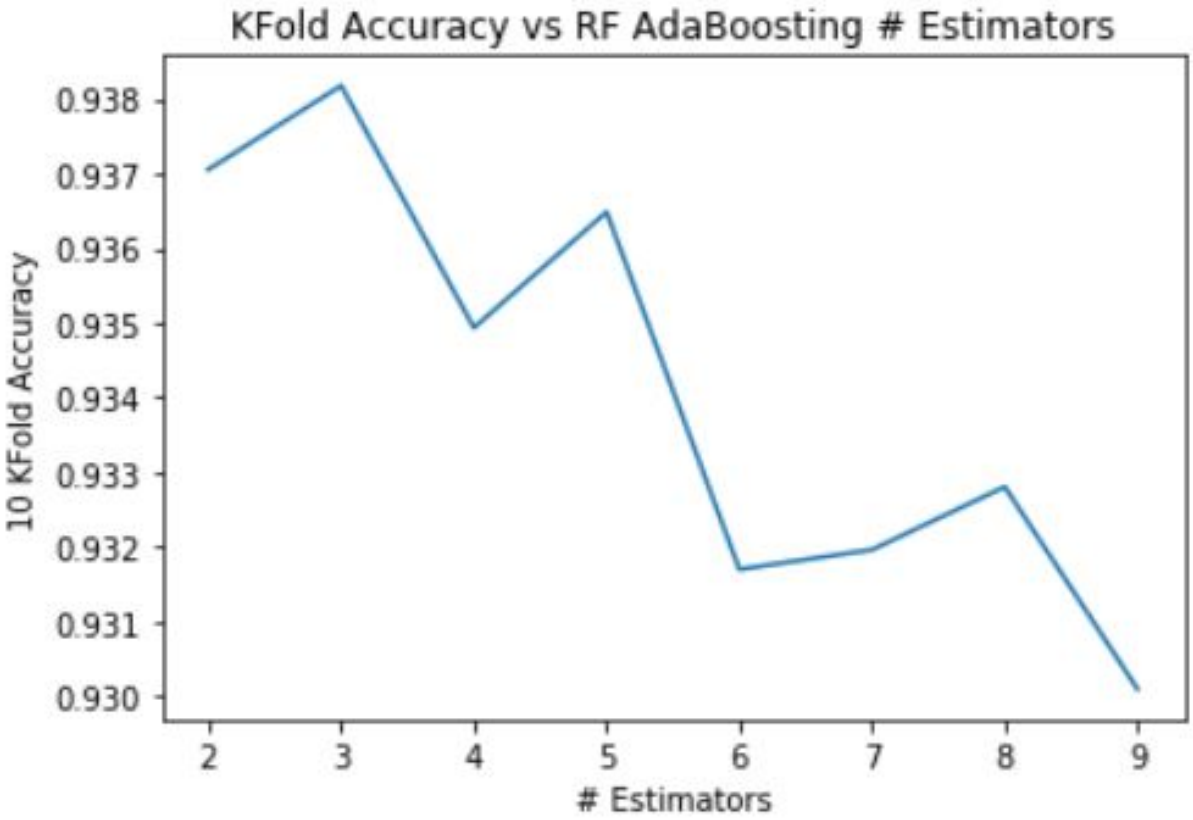

Figure 7: Random Forest, Ada-Boosting

In the above graph we use the random forest for the thigh sensor to do our Ada-Boosting grid search. As stated before, we are using the thigh sensor because so far it has shown to be the best sensor for prediction accuracy. As seen in the above graph, Ada-Boosting a random forest is largely ineffective in improving prediction accuracy. We decided to grid search over a smaller scale (2-10 estimators) because we wanted to see if just a small number of Ada-Boosted models could increase accuracy. We also suspect that Ada-Boosting was

ineffective in this application due to the already high accuracy of our random forest, which started at an accuracy of 94.1% (see table 17).

7.6 Decision Tree

For our next model we used a decision tree for classifying final activity. This classifier was actually one of the last models we built near the end of the capstone. Due to the lack of time, we left the model parameters as the defaults provided by Sklearn's decistion tree package. Building the decision tree with defaults isn't necessarily bad as most of the decision tree parameters by default are already hypertuned algorithmically. With this known, there are some assurances that the default parameters for the decision tree are reasonable.

7.7 Segmentation and Neural Network

To better capture the periodic nature and high variance of some activities, we segmented the data before classifying it. This segmentation step allowed us to classify variable length bouts of activities, as opposed to classifying each second of data individually.

7.7.1 Segmentation Implementation

To segment our data, we utilized change point detection algorithms from the `Ruptures` Python library. In particular, we utilized their implementation of the PELT method, as it is both exact—meaning it will find the optimal change points—and does not require the number of segments to be provided. These aspects of the PELT method make it the most flexible while adapting it for our purposes.

While we are not required to provide the number of segments we want as a parameter, there is a penalty parameter that must be optimized to ensure a balance between bout accuracy and length. This is similar to training a machine learning model, so we chose to perform an exhaustive search over a large range of penalty values.

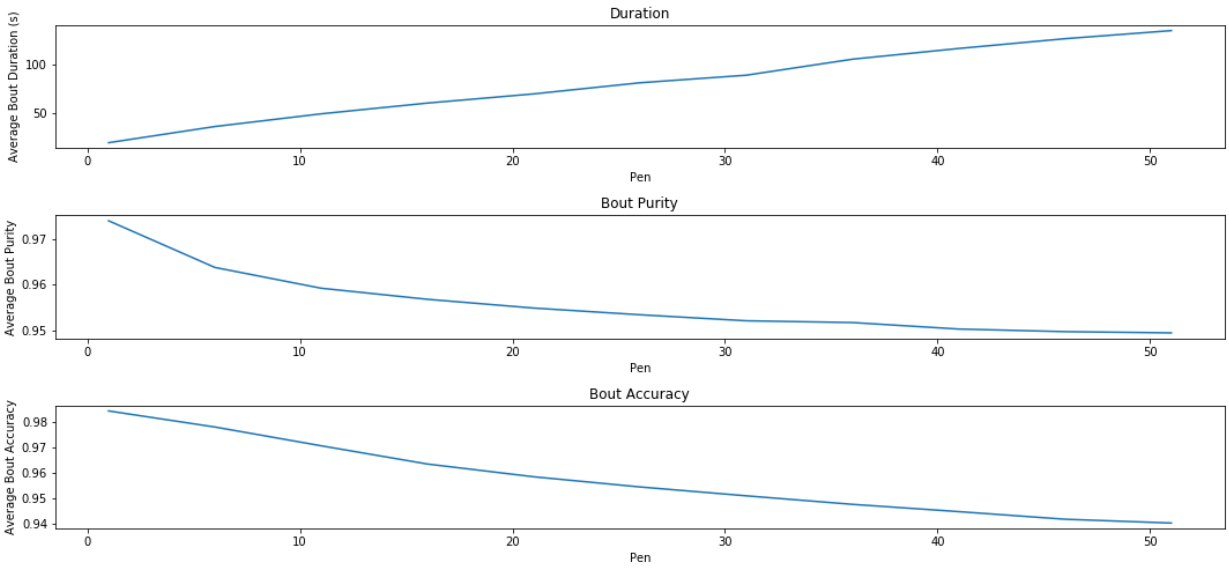

Figure 8: Penalty Value Search

As seen in Figure 8, we used three metrics in our penalty value search: average bout duration, purity, and accuracy. Bout duration is how many seconds there are between two change points, bout purity is the proportion of the bout that is the mode, and bout accuracy is how accurate our bouts are to the original if the entire bout is given the label of the mode.

Figure 8 shows that the average bout duration increased linearly as the penalty increased, while the average bout accuracy decreased linearly as the penalty increased. The average bout purity had a sharp dip initially, but followed the same trend as the other two metrics afterward. This parameter search indicates that there is no obvious optimal penalty value. As a result, we decided based on trying to maximize duration while ensuring that we do not lose too much accuracy.

7.7.2 Segmentation Results

The segmentation algorithm was able to identify change points in our data with a high degree of success. The predicted change points aligned closely with those from the original data.

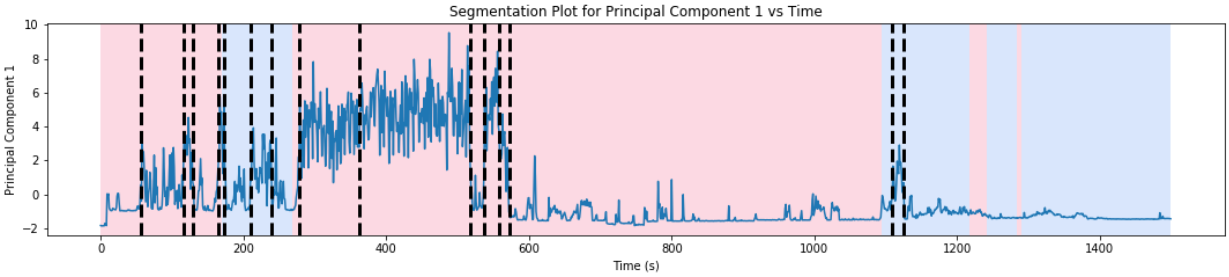

Figure 9: Example of Segmented Data

In Figure 9, the change of background color indicates the true activity bouts, while the dotted black lines are the change points that our algorithm predicted. As seen in Figure 9, while we predict far more change points than are necessary, we are able to accurately distinguish the true changes in activity.

7.7.3 Classification

To factor in our newly segmented bouts into our classification model, we had to transform the data from each bout into a form that our neural networks could process. We opted to represent each bout as a single observation and created summary statistics for each bout by taking the mean and standard deviation of each feature. This essentially doubles our original feature space.

8 Results and Conclusion

8.1 Accuracy Metrics

The accuracy metrics that will be used to determine model strength are explained below.

8.1.1 Terms to Know

Our accuracy metrics are built on the following terms. These terms will be put into context of classifying the "running" activity as a concrete example.

- True Positive (TP) - The number of data points classified correctly as "running".
- False Positives (FP) - The number of data points incorrectly classified as "running" when they were actually another activity.
- False Negatives(FN) - The number of data points that were actually "running" but classified as another activity

8.1.2 Precision

Precision = TP / (TP + FP)

Precision is a measure of a model's ability to identify only the relevant data points. For example a model that classifies many "running" data points correctly, but also mis-classifies many datapoints that are not "running" as "running" will have low precision. If this model classified many "running" data points correctly and also avoided mis-classifying other activities as "running" the model will have high precision.

8.1.3 Recall

Recall = TP / (TP + FN)

Recall is a measure of a model's ability to find all relevant cases within a dataset. For example a model that classifies almost all datapoints that are actually "running" as "running" will have high recall. Conversely, a model that classifies some datapoints that are actually "running" as "running" and miscassifies many datapoints that are actually "running" as another activity will have low recall.

8.1.4 Overall Accuracy (Microlevel)

The overall accuracy for each model will be calculated on a microlevel for the Kfold cross validation. This means that for n Kfolds we will sum the number of data points predicted correctly over the total number of data points predicted.

8.2 Single Sensor Neural Network

Below are the results for four neural networks, one for each sensor.

Table 7: Single Sensor Neural Network: Chest Final Activity

		Predicted					
		Sit/Lie	Stand and Move	Walking	Running	Bicycle	**Recall**
Actual	Sit/Lie	148246	32019	114	3	181	**0.821**
	Stand and Move	23956	102660	5178	386	281	**0.775**
	Walking	77	9543	24110	58	8	**0.713**
	Running	66	857	51	8031	244	**0.868**
	Bicycle	1125	1018	0	13	3952	**0.647**
	Precision	**0.855**	**0.703**	**0.819**	**0.946**	**0.847**	**0.792**

Table 8: Single Sensor Neural Network: Hip Final Activity

		Predicted					
		Sit/Lie	Stand and Move	Walking	Running	Bicycle	**Recall**
Actual	Sit/Lie	138618	28771	157	2	139	**0.827**
	Stand and Move	14027	107996	5067	266	418	**0.845**
	Walking	26	9318	20846	114	72	**0.686**
	Running	1	1798	252	7147	0	**0.777**
	Bicycle	112	5708	286	1	1	**0.0**
	Precision	**0.907**	**0.703**	**0.783**	**0.949**	**0.002**	**0.805**

Table 9: Single Sensor Neural Network: Thigh Final Activity

		Predicted					
		Sit/Lie	Stand and Move	Walking	Running	Bicycle	**Recall**
Actual	Sit/Lie	177676	2223	173	1	303	**0.985**
	Stand and Move	840	123163	7671	456	330	**0.93**
	Walking	2	7758	25799	146	91	**0.763**
	Running	8	402	156	8648	35	**0.935**
	Bicycle	388	998	234	14	4474	**0.732**
	Precision	**0.993**	**0.915**	**0.758**	**0.933**	**0.855**	**0.939**

Table 10: Single Sensor Neural Network: Wrist Final Activity

		Predicted					
		Sit/Lie	Stand and Move	Walking	Running	Bicycle	**Recall**
Actual	Sit/Lie	121641	45360	533	37	116	**0.725**
	Stand and Move	42742	79752	4153	528	599	**0.624**
	Walking	1853	15942	12124	109	348	**0.399**
	Running	9	1867	92	7205	25	**0.783**
	Bicycle	100	5020	846	142	0	**0.0**
	Precision	**0.731**	**0.539**	**0.683**	**0.898**	**0.0**	**0.647**

Looking at the above results we see that the thigh sensor yields the highest accuracy overall and the best precision/recall results for all classes. Looking closer we can see where other sensors such as the wrist and hip are doing poorly. For example the wrist and hip sensor do poorly for classifying "bicycling". In context this makes sense considering the location of the hip and wrist during a bicycling activity. Wrists are mostly stationary while holding the handlebars and the hip is also relatively stationary while sitting on a bike. A thigh sensor on the other hand would better catch a bicycling activity due to the up and down movement of pedaling. This analysis is important because we can see here that the sensor we used to train our models does matter.

8.3 Two Sensor Neural Network

Below are the results for all two sensor neural networks, one for each two sensor combination.

Table 11: Two Sensor Neural Network: Chest & Wrist Final Activity

		Predicted					
		Sit/Lie	Stand and Move	Walking	Running	Bicycle	**Recall**
Actual	Sit/Lie	124672	42713	98	4	64	**0.744**
	Stand and Move	26981	97311	3061	222	55	**0.762**
	Walking	376	13759	16175	51	15	**0.532**
	Running	14	806	47	8152	179	**0.886**
	Bicycle	966	4892	154	88	8	**0.001**
	Precision	**0.815**	**0.61**	**0.828**	**0.957**	**0.025**	**0.723**

Table 12: Two Sensor Neural Network: Hip & Chest Final Activity

		Predicted					
		Sit/Lie	Stand and Move	Walking	Running	Bicycle	**Recall**
Actual	Sit/Lie	140661	26751	82	1	56	**0.84**
	Stand and Move	10117	113092	4095	203	123	**0.886**
	Walking	176	9007	21139	38	16	**0.696**
	Running	16	814	147	8221	0	**0.894**
	Bicycle	835	5055	5	26	187	**0.031**
	Precision	**0.927**	**0.731**	**0.83**	**0.968**	**0.49**	**0.831**

Table 13: Two Sensor Neural Network: Hip & Thigh Final Activity

		Predicted					
		Sit/Lie	Stand and Move	Walking	Running	Bicycle	**Recall**
Actual	Sit/Lie	164742	2505	67	3	47	**0.984**
	Stand and Move	1335	121573	4448	183	90	**0.953**
	Walking	4	10060	20195	109	8	**0.665**
	Running	2	249	247	8700	0	**0.946**
	Bicycle	1255	3744	83	66	960	**0.157**
	Precision	**0.984**	**0.88**	**0.807**	**0.96**	**0.869**	**0.928**

Table 14: Two Sensor Neural Network: Hip & Wrist Final Activity

		Predicted					
		Sit/Lie	Stand and Move	Walking	Running	Bicycle	**Recall**
Actual	Sit/Lie	140711	26807	114	2	53	**0.839**
	Stand and Move	20910	102530	3791	185	358	**0.802**
	Walking	347	14605	15336	67	21	**0.505**
	Running	13	2456	494	6233	2	**0.678**
	Bicycle	149	5853	105	0	1	**0.0**
	Precision	**0.868**	**0.673**	**0.773**	**0.961**	**0.002**	**0.776**

Table 15: Two Sensor Neural Network: Thigh & Chest Final Activity

		Predicted					
		Sit/Lie	Stand and Move	Walking	Running	Bicycle	**Recall**
Actual	Sit/Lie	177937	2266	81	0	92	**0.986**
	Stand and Move	2137	124137	5544	386	252	**0.937**
	Walking	3	8503	25214	63	13	**0.746**
	Running	13	1153	102	7968	13	**0.861**
	Bicycle	1121	1200	4	9	3774	**0.618**
	Precision	**0.982**	**0.904**	**0.815**	**0.946**	**0.911**	**0.937**

Table 16: Two Sensor Neural Network: Thigh & Wrist Final Activity

		Predicted					
		Sit/Lie	Stand and Move	Walking	Running	Bicycle	**Recall**
Actual	Sit/Lie	164883	2252	114	7	108	**0.985**
	Stand and Move	993	121169	4923	376	168	**0.949**
	Walking	3	16002	14163	127	81	**0.466**
	Running	23	1419	146	7461	149	**0.811**
	Bicycle	518	1730	2450	268	1142	**0.187**
	Precision	**0.991**	**0.85**	**0.65**	**0.906**	**0.693**	**0.906**

From our results we can see that combining sensors at least for our neural network implementation does not give us noticeable increases in accuracy. The best two sensor neural network combines the thigh and chest data (see Table 15), but even then the overall accuracy is just even with the single sensor thigh accuracy.

8.4 Random Forest Classifier

The results for the random forest classifier are below. At this point in the capstone our team decided to use only the thigh sensor for the remaining random forest classifier and decision tree classifier . The justification for this being that training and hyper-tuning random forests/decision trees for each sensor is expensive computation wise. Secondly, as our single sensor neural networks showed, the thigh sensor showed notably better accuracy results for all final activity classes and overall accuracy. With limited resources and time, our team focused our efforts on the thigh sensor which has given us promising results so far.

Table 17: Random Forest: Thigh Final Activity

		Predicted					
		Sit/Lie	Stand and Move	Walking	Running	Bicycle	**Recall**
Actual	Sit/Lie	177951	2239	136	0	50	**0.987**
	Stand and Move	1029	123845	7166	304	116	**0.935**
	Walking	3	7241	26485	26	41	**0.784**
	Running	0	102	136	9005	6	**0.974**
	Bicycle	327	787	1600	11	3383	**0.554**
	Precision	**0.992**	**0.923**	**0.746**	**0.964**	**0.941**	**0.941**

8.5 Decision Tree Classifier

The results for the decision tree classifier are below.

Table 18: Decision Trees: Thigh Final Activity

		Predicted					
		Sit/Lie	Stand and Move	Walking	Running	Bicycle	**Recall**
Actual	Sit/Lie	175934	4082	164	6	190	**0.975**
	Stand and Move	3595	120128	8041	396	300	**0.907**
	Walking	359	12961	20148	202	126	**0.596**
	Running	2	618	84	8529	16	**0.922**
	Bicycle	578	1959	669	60	2842	**0.465**
	Precision	**0.975**	**0.86**	**0.692**	**0.928**	**0.818**	**0.905**

8.6 Segmentation and Neural Network

The results for the segmentation and neural network model are below.

Table 19: Segmentation: Chest Final Activity

		Predicted					
		Sit/Lie	Stand and Move	Walking	Running	Bicycle	**Recall**
Actual	Sit/Lie	180563	0	0	0	0	**1.0**
	Stand and Move	132461	0	0	0	0	**0.0**
	Walking	33796	0	0	0	0	**0.0**
	Running	9249	0	0	0	0	**0.0**
	Bicycle	6108	0	0	0	0	**0.0**
	Precision	**0.499**	**0.0**	**0.0**	**0.0**	**0.0**	**0.499**

Table 20: Segmentation: Hip Final Activity

		Predicted					
		Sit/Lie	Stand and Move	Walking	Running	Bicycle	**Recall**
Actual	Sit/Lie	155747	11688	30	0	222	**0.929**
	Stand and Move	12448	111348	3653	229	96	**0.871**
	Walking	16	6325	23873	78	84	**0.786**
	Running	20	351	404	8423	0	**0.916**
	Bicycle	75	5448	555	30	0	**0.0**
	Precision	**0.925**	**0.824**	**0.837**	**0.962**	**0.0**	**0.878**

Table 21: Segmentation: Thigh Final Activity

		Predicted					
		Sit/Lie	Stand and Move	Walking	Running	Bicycle	**Recall**
Actual	Sit/Lie	177928	2337	51	0	60	**0.986**
	Stand and Move	1180	126463	4393	299	125	**0.955**
	Walking	14	7339	26363	68	12	**0.78**
	Running	0	478	75	8690	6	**0.94**
	Bicycle	245	730	672	6	4455	**0.729**
	Precision	**0.992**	**0.921**	**0.835**	**0.959**	**0.956**	**0.95**

Table 22: Segmentation: Wrist Final Activity

		Predicted					
		Sit/Lie	Stand and Move	Walking	Running	Bicycle	**Recall**
Actual	Sit/Lie	136121	31359	145	0	62	**0.812**
	Stand and Move	39519	83547	3903	343	462	**0.654**
	Walking	888	8860	19849	736	43	**0.653**
	Running	0	1051	289	7852	6	**0.854**
	Bicycle	62	5575	423	48	0	**0.0**
	Precision	**0.771**	**0.641**	**0.807**	**0.875**	**0.0**	**0.725**

As seen in Table 19, once we segment our data and aggregate our bouts, our neural network is incapable of distinguishing the difference between activities, and thus naively classifies the entire dataset as "sit/lie." The loss in accuracy for the chest sensor is contrasted by accuracy increases across the remaining sensors compared to the single sensor neural network results. The hip and wrist sensors both had close to an 8% boost in accuracy, and the thigh sensor had a 1% boost in accuracy.

9 Conclusion

As illustrated in our results, our segmentation and neural network model performed the best for each sensor individually, and achieved our best overall accuracy of 95% for the thigh sensor. While the results of the segmentation model are promising, it is time consuming to optimize the segmentation algorithm and process the data with it. We are also doubling our feature space between the segmentation and classification steps, which compounds this issue. The random forest classifier is able to achieve an accuracy of 94% for the thigh sensor in significantly less overall runtime.

References

[1] Lyden, Kate et al. A method to estimate free-living active and sedentary behavior from an accelerometer Medicine and science in sports and exercise vol. 46,2 (2014): 386-97.

[2] Staudenmayer, John et al. Methods to estimate aspects of physical activity and sedentary behavior from high-frequency wrist accelerometer measurements Journal of applied physiology (Bethesda, Md. : 1985) vol. 119,4 (2015): 396-403.

[3] Willetts, M., Hollowell, S., Aslett, L., Holmes, C. and Doherty, A. (2018). Statistical machine learning of sleep and physical activity phenotypes from sensor data in 96,220 UK Biobank participants. Scientific Reports, 8(1).

[4] Klein: Machine Learning- Boosting,
`https://www.python-course.eu/Boosting.php`

10 Appendix

Feature	**Description**
min.x, min,y, min.z	Minimum x, y, and z acceleration in interval
max.x, max.y, max.z	Maximum x, y, and z acceleration in interval
mean.x, mean.y, mean.z	Mean of x, y, and z acceleration in interval
mean.xy, mean.xz, mean.yz, mean.xyz	Mean products of x, y, and z accelerations in interval
sd.x, sd.y, sd.z	SD of x, y, and z acceleration in interval
mean.vm	Mean of the vector magnitude in interval ($\text{vm} = \sqrt{x^2 + y^2 + z^2}$)
sd.vm	SD of the vector magnitude in interval
pct25, pct75	25th and 75th percentiles of the vector magnitude in interval
mean.ang	Mean angle of acceleration relative to vertical ($\text{angle} = 90 arcsin(x/vm)/(\pi/2)$)
sd.ang	SD of the angle of acceleration relative to vertical
xfft, yfft, zfft, mfft (1-15)	Fast Fourier Transform (1-15 Hz) of the x, y, and z accelerations, and the vector magnitude (mfft)

Table 23: Features and Descriptions

Activity Domain	Description
Caring for Others	Caring for and helping adults, caring for and helping children
Eating	Eating and drinking or waiting
Education	Taking class/research/homework
Exercise	Attending sport/recreational event/performance, baseball, frisbee, jogging, other, walking
Housework	Animals and pets, exterior maintenance/repair/decoration, food prep and cleanup, household management/other household activities, housework, interior maintenance/repair/decoration, lawn/garden/houseplants
Leisure	Screen based leisure time (TV, video game, computer), socializing/communicating/leisure time not screen
Organizational	Organizational civic/volunteer/religious activities
Other	Non codeable
Personal Care	groom/health-related, other personal care
Purchasing	Purchasing goods and services
Travel	Biking, driver (car/truck/motorcycle), passenger bus or train, walking
Work	General - education and health services, general - office (business, professional services, finance, info), screen based - education and health services, screen based - office (business, professional services, finance, info)

Table 24: Activity Per Second Ground Truth Encoding

Posture	Description
Leisure	kneeling/squatting, stand, stand and move, stand and move with unidentifiable upper body movement, stand and move with upper body movement, stretching
Sedentary	lying, sitting
Sport	bike, other sport movement
Walking	ascend stairs, descend stairs, running, walk, walk with load

Table 25: Posture Per Second Activity Encoding

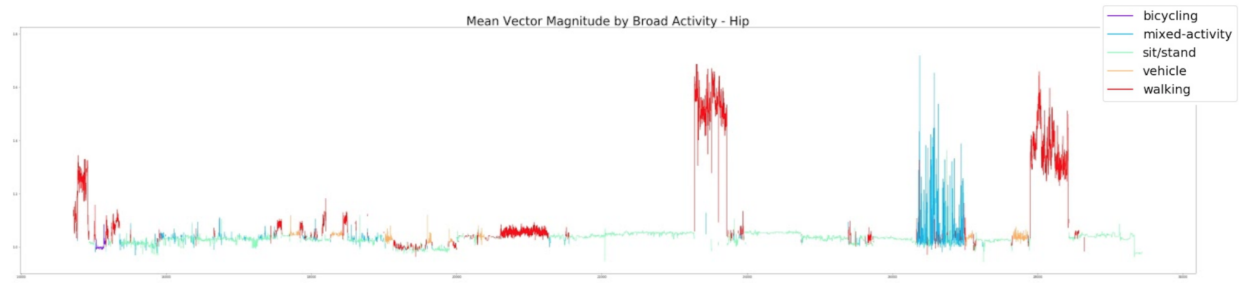

Figure 10: Mean Vector Magnitude by Time, Colored by Broad Activity - Hip

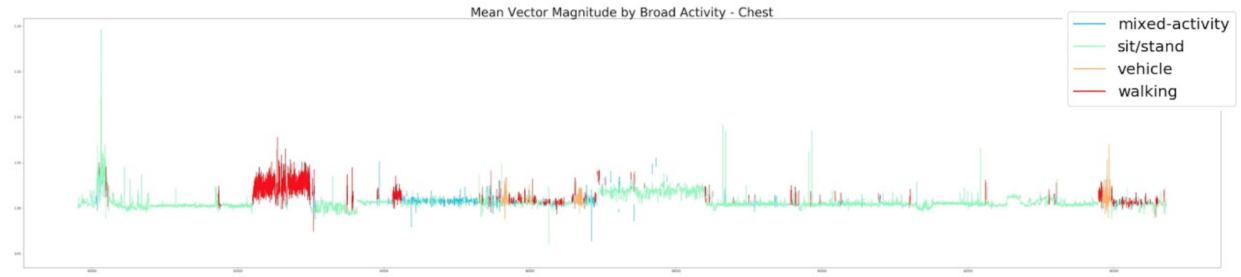

Figure 11: Mean Vector Magnitude by Time, Colored by Broad Activity - Chest

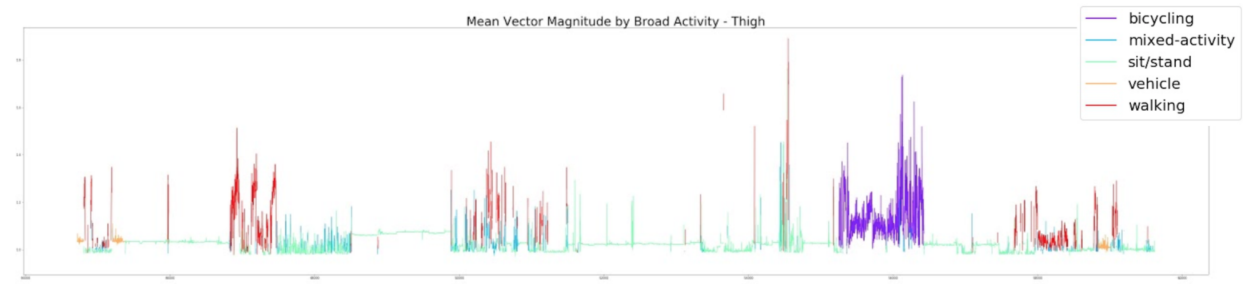

Figure 12: Mean Vector Magnitude by Time, Colored by Broad Activity - Thigh

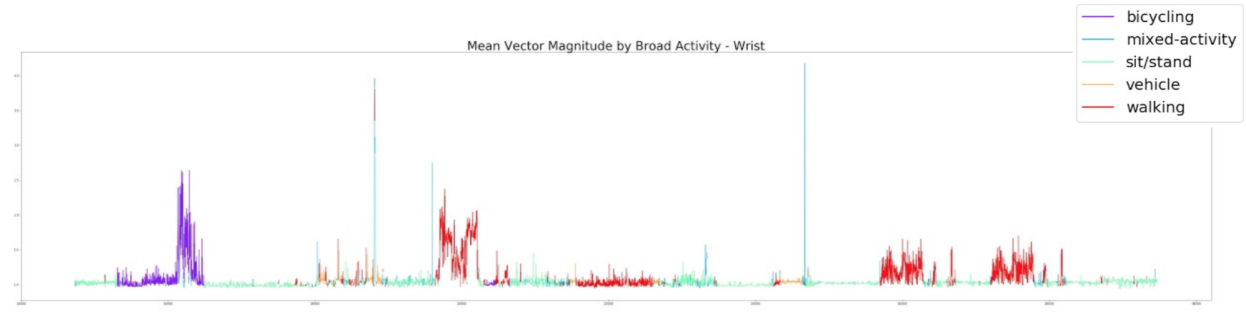

Figure 13: Mean Vector Magnitude by Time, Colored by Broad Activity - Wrist

Figure 14: Legend for Figure **??**

Figure 15: Legend for Figure **??**