FitFeet Smart Shoe System

Dylan Vanmali
School of Electrical and Computer Engineering
Carnegie Mellon University
Email: dvanmali@andrew.cmu.edu

Chingyi Lin
School of Electrical and Computer Engineering
Carnegie Mellon University
Email: chingyil@andrew.cmu.edu

***Abstract*—An algorithm for developing a smart shoe sole is outlined. This application is intended as a health monitoring system that would generate valuable data involving athletic abilities like walking and running. The device is able to understand walking patterns better than current market solution due to its location on the body.**

An embedded sensor on your feet gives unique data which traditional sensors at their current locations on wrists or inside pockets are unable to provide. By integrating a collection of pressure sensors, positioning sensors, and wireless communications to an online user interface, we are able to offer both health care recommendations and activity monitoring solutions.

I. Introduction

By better understanding the positioning and stress applied on feet with respect to the body, we hope to better classify physical activity. Our FitFeet smart shoe solution enables users to understand their walking patterns with higher precision and enables us to provide recommendations for incorrect moving patterns before future injuries ensue.

Using a network of pressure sensors around the bottom of the foot and Inertial Measurement Units (IMUs) on the top of the foot, users can see daily statistics that we gather about their movement around the environment. We can then classify each kinetic movement to determine whether or not their daily movement patterns are healthy, otherwise we notify them of possible solutions to improve their behavior, such as urging them to exercise or to seek medical advice. By offering more personalized physical monitoring and recommendation systems, the FitFeet smart shoe enables users to accurately understand their physical activity.

Ultimately, FitFeet serves as a more accurate activity monitor by classifying activity levels like stepping, running, walking, jumping, and standing. This allows us to better segment movement types throughout the day and provide them to a user with this exact precision. Beyond this, we hope to categorize more complex behavior such as injury, tiredness, swaying, jittering, and duck-walking, and then provide feedback so that the user is more aware of their body.

II. Previous Work

Existing market solutions target mobility away from the feet and rather on the wrist. Brands like Fitbit and the Apple watch generate approximate stepping data targeting athletes and tech enthusiasts.

On the other end of the spectra, companies that focus on feet pressure sensing, such as the French company FeetMe, target recovering patients by providing them with biometric data about their weight distribution patterns. Another shoe company, E-Vone, serves as a lifesaving solution for elderly members by detecting their falling patterns.

Beyond health, smart shoes exist in athletic apparel such as IOFIT Smart Shoes which has the ability to help golfers understand their weight distributions while swinging. Even companies like Bolt and Nike focus on running analytics to enable athletes to achieve peak performance.

To best of our knowledge, there does not exist any shoe company that combines performance analytics, movement classification, and recommendation services so health-conscious consumers are more aware of their physical activity. With the growing movement for more active lifestyles, FitFeet would offer the perfect solution to understanding both short-term classification and long-term patterns.

Fig. 1. The main Dashboard with the ultimate statistics such as activity classification gathered over time and displayed with statistics results throughout the day.

III. Approach

A. Sensing target and domain

As a personalized activity monitoring system, FitFeet aims to identify between the following types:

1) Step Counting: Since a step is determined as a burst of force between the ground and the sole of the foot, FitFeet can use its pressure mappings to provide a more accurate step count as it notices this change through time. Here we provide users daily goals to achieve and motivate users to achieve them throughout the day.
2) Jumping: A jump is a large vertical acceleration spike with a large delay between an initial standing position to a jolt upon returning to the ground. Using the accelerometer and pressure sensors, we can enable this performance metric independently from the average step.
3) Running and Walking: Both the speed at which the user accelerates forward and the rate between each step is needed to individualize running, walking, and stationary movements from each other, but also from other motion within cars or planes. FitFeet maps each agility on a timeline in order to see how much of each activity is performed every day.
4) Inactivity: Understanding whether the user has moved away from a specific location is done by measuring no changes in the user's state or steps. In other words, when no shifts in position are signaled, the user is considered to be inactive. Thus, we want to provide a reminder that the user should get up and move around the space throughout their day. Standing and sitting are subsets in this category as the user does not move away from their initial position.

For more complex behaviors, FitFeet aims to distinguish between various motion patterns and alert the user if the system senses these abnormalities:

1) Injury: People who may be injured will have an irregular walking patterns. Most of them may ignore these injuries because their mind will find a way to move the stress away from that injured point applying pressure to other areas of the foot. FitFeet aims to detect that irregularity and notify the user about their potential injury.
2) Tiredness: Physical fatigue is reflected by a slow body motion or deep steps over long periods of time. For example, people working long hours might feel exhausted and not move around the space as much. In this case, we want to recommend the user to sleep.
3) Swaying: A swaying motion is classified as a motion from left to right while moving forward. For example, people who have drank too much alcohol typically exhibit this behavior. By figuring out this motion, we can recommend the user to not drive.
4) Jittering: Known as Restless Leg Syndrome, a fast upwards and downwards motion could commonly indicates a leg jitter. FitFeet aims to notify the user to become more aware of this subconscious behavior and fix it if they wish.
5) Duck-Walk: A walking posture where the feet point away from the forward motion of the user. FitFeet can compare its magnetometer reading on each foot to indicate whether the user is walking forward. If the magnetometer readings point in different directions, then this may be an indication of an incorrect walking pattern so FitFeet can notify the user.
6) Falling: A depressurization of the foot combined with a gyroscopic reading that shows the user lies horizontally or frontward could be an indication for a fall. Thus, we can alert the user or immediate family for assistance.

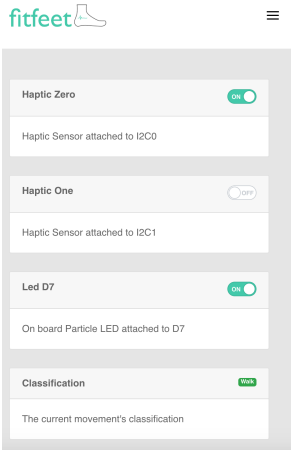


Fig. 2. The Admin web-page that allows manual control of the Haptic Controllers, the on-board LED, and a real-time visual the Machine Learning classification interpretation.

B. Frequency domain analysis

Frequency domain analysis has been proven for its successes in various domains such as voice recognition or ECG signal processing. In FitFeet, we also apply the frequency domain analysis for the sensor signal. A fixed-length of signal in time domain will be transformed into a series of frequency-domain components. The intuition here is based on the ubiquitous frequency characteristic in different moving behavior. For example, slow behavior like walk has a peak in some low-frequency components. On the other side, intense behavior such as running shows higher magnitude in high-frequency components.

The transformation is achieved by Fast Fourier Transform (FFT), which optimizes discrete Fourier Transform (DFT) on CPU. At each timestep t, a fixed m-length signal from x[t-m] to x[t] is evaluated by FFT, and we can obtain a complex m-vector f_t at current time t. This transformation converts a sequence of time-domain signal into another shorter sequence of frequency-domain signal. In the following section, we are going to leverage those frequency components to complete the classification task.

Before the start of classification under frequency domain, we need to make sure that FFT helps us in our classification task. We use Principal Component Analysis (PCA) to project the signal into two-dimensional space. Figure 3 shows the

original distribution under the time domain, three behaviors walk, run, inactive are scattered in an xy-plane and colored with red, blue, green respectively. Two axes in this figure show the projected axis after the transformation from PCA. From this figure, we found that it's hard to tell each data point from its respective behavior noticing that red and blue points are mixed together while the green dots are isolated in the scatter.

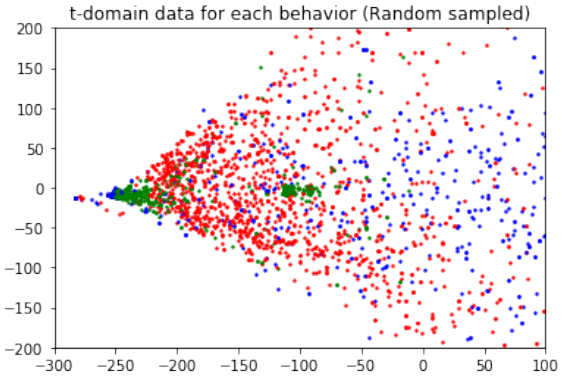


Fig. 3. Time-domain visualization

After we apply FFT, the scatter from PCA shows a more easily classified scatter on the x-y plane. Shown in Figure 4, it is more obvious that a decision boundary in the x-y plane can be drawn to segment them into behavior classes. While it is not perfect, the green dots represents inactive concentrates in a small area, showing less variance in frequencies in overall. On the other end, it is easy to observe the cluster zones for both the red and blue data points after the FFT is applied.

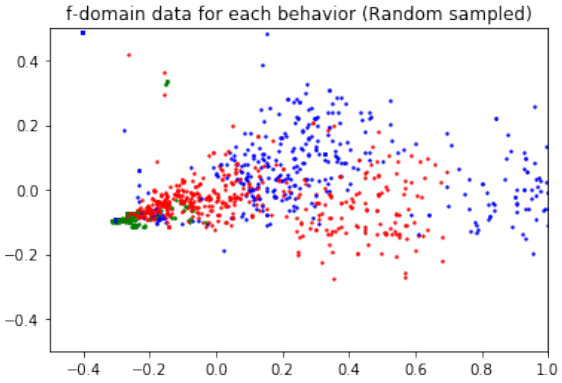


Fig. 4. Frequency-domain visualization

C. Support Vector Machine in frequency domain

As a classic machine learning algorithm, support vector machine (SVM) has shown its power in numerous problems. Its good performance on higher dimensional space and high accuracy in both binary or multiple class classification have shown its success.

Our classification task is: Given a m-vector frequency component, classify it into a behavior b, where $b \in \{b_0, b_1, ;b_k\}$, k is the total number of behaviors in our targets. This problem matches the scenario of multi-classes SVM. We are able to take a vector from FFT, and classify it as one of the most likely behavior in our pool.

The classifier program will collect a fixed number of preprocessed data points in the frequency domain. Those frequency components run throught the SVM with linear kernel and get the prediction from the highest score in one-vs-rest (OVR) decision function. In this training task, we didn't do dimensional reduction since this application doesn't have a latency constraint, also we don't want to sacrifice the accuracy. The normalization in preprocessing prevents the scaling effect in our SVM, which will be discussed in the next section along with the kernel selection. We also enable the probability calculation in our SVM, to show our confidence for each classification task.

D. Website

The data-visualizer used to display our results is run using languages like HTML, CSS, JS, and JQuery and calls many APIs such as bootstrap for responsive dynamic layouts, chart.js for the graphs, and Paho and Mosquitto for MQTT.

In order to not overwhelm our long-term server and create a scalable system that does not break our interactions, the Website only updates its values every two seconds. While the data to be stored is reacted immediately when data can be process, we restrict this half of the system to not overwhelm the server and the user can still obtain almost immediate response time back.

E. Data storage with circular buffer

A sub-sequence of frequency components are required from the analysis with frequency-domain SVM. These most recent n data points from sensor should be stored into some data structure. Theoretically, we can create a dynamic array and put all the data in one sequence. With the pointer of the last element, we can extract the sub-sequence consists from x[t-n] to x[t]. However, this dynamic array increases the storage size as more and more data are collected. This poor scalability uses the storage inefficiently and makes it impractical.

We leverage circular buffer instead of a dynamic array. With pre-defined length n, the circular buffer stores the data points in order from the first element. Once the counter overflows, the pointer goes back to the start of data. This technique utilize the limited storage in a more efficient way, which preserves the most recent data and release the usage for those out-dated data.

Under circular buffer, extracting the last n data points is easy. Since n is already known, we can define this n as the length of the circular buffer. In the n-length circular buffer, all the elements are required in analysis. The order begins from the current pointer to the last, and restart from the start to the element before the pointer.

F. MQTT protocol

Connecting all the components above, MQTT protocol is used as the communication. Its publication/subscription model

gives convenience for the micro-controller and classification program to exchange the data. The dataflow will be discussed in the next section.

IV. Experimental Setup and Results

A. System Overview

Our system consists of a micro controller (Particle Argon) attached to the IMU sensor previously mentioned, a local classification server, and an user interface used to display important user behavior. All of these components are connected between each other using an MQTT relay broker which handles where the data needs to be sent.

The data of our device begins where the sensors get the measured signal from our FitFeet device. The micro controller then uses MQTT protocol over a WiFi antenna to send its raw data to the classification server. After the classification completes using the frequency-domain SVM algorithms, the prediction and the probability of that prediction is then sent to the long term-storage center which keeps the data for future use whenever needed. Acting like a database, this server is specialized using MQTT to input data in time frames based on classification. Thus while the entire system sees current data classifications, this long-term storage allows us to generate statistics catered to the user's needs. Finally, the display can then access any live or long-term data simply by connecting to the MQTT broker that communicates between all these devices.

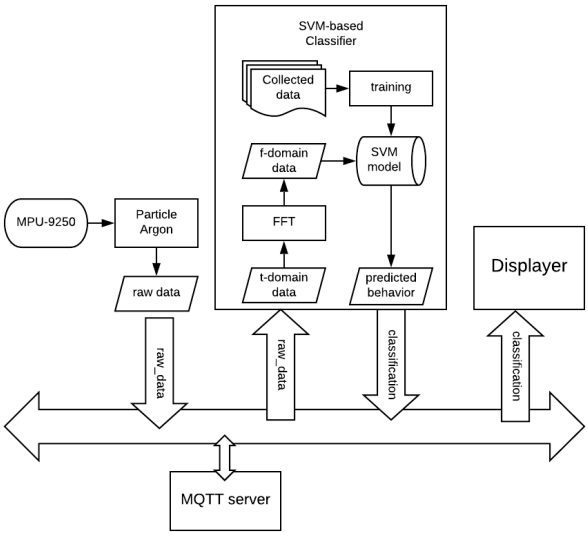


Fig. 5. System overview

B. Sensors selection

1) 9-axis IMU sensor: Using its built-in 3-axis gyroscope, 3-axis accelerometer, and 3-axis magnetometer, our IMU, the MPU-9250, can help us obtain lots of useful physical data. With the accelerometer, we can see the speed at which the user is moving. With the gyroscope, we can track the orientation of the foot in space. With the magnetometer, we can tell the intended direction of walking through the world.

Currently, our system can receive data from a single IMU on the server and display the stream of values on the Notifications web-page. This achievement is important because the system needs to take this upstream of data from the device and perform its high-intense computations on the server.

Fig. 6. FitFeet Shoe on a printed circuit board with attached battery, IMU, and Haptic controller attached

2) Force-Sensitive Resistor: A Force-Sensitive Resistor (FSR) is a great way to collect points of pressure throughout the foot. By mapping multiple pressure points at known positions, we can then generate an approximate map personalized towards that person's foot weight distribution. The magnitude of each pressure point is simply determined as a ratio between the varying resistor and a specified resistor. Each FSR can support up to 20 lbs (about 100 Newtons) each and we can read this value as an analog value so we can approximate whether a step has contact with the ground or is above the air. While a normal user weighs considerably more than 20 lbs, the distribution of these points at various locations gives us the variability of contacts points needed to better determine a physical step. It is not important to determine the user's weight at a given point but rather to understand whether a binary result of a force-applied or force-not-applied was taken at each contact point: two FSRs in the back of the foot and two FSRs towards the front of the foot.

Currently, our system can receive data from all pressure sensors on the server and display the stream of values on the Notifications webpage. This achievement is important because users favor visualizing data, and this allows the webpage to display important statistics, as an upstream from the device to the server.

3) Haptic motor: Since walking is an unconscious behavior, we attempt to alert our user's walking patterns in a similar matter. Through conditioning, FitFeet provides haptic feedback beyond the visual feedback. This allows a faster approach in targeting the issue while it occurs. Our walking behavior should be corrected in a short time since changing

your walking patterns are not intentional. To drive a haptic motor, we use a haptic motor controller DRV2605L. With I2C interface, Particle Argon can send the control signal on I2C bus and trigger the haptic feedback to the user.

Currently, our system can signal the device from the Admin webpage and manually turn on the Haptic Motor. This achievement is important when the server needs to signal the user for incorrect behavior after running its algorithm, in other words a downstream from the server to the device. We are able to trigger the Haptic response back to the user whenever a jitter motion is classified, which can later be applied to any other feedback notification response previously mentioned when other classification categories become available.

4) Battery: As a wearable device, portable power supply is a factor to a practical product. We have chosen a 3.7V 2000mAh Li-Po battery instead of a typical power bank because it is both thin (6.6mm) and light-weight (39g), thus providing good power to power our circuit board in a small form-factor.

C. Local MQTT server

The question occurs about where should the various components of this system compute its data. Using IoT edge computing methods, FitFeet aims to compute this data as efficiently as possible to obtain close to real-time behavior.

By default, the Particle device has a cloud server that enables users to publish and subscribe to obtain and receive data, respectively, from edge devices. However, as a moving pattern classification, sampling frequency is a hard constraint in our application. With a 2-packet-per-second bandwidth limitation from Particle Cloud Server, the default system model cannot satisfy the speed requirement for our application.

MQTT is a well-known lightweight protocol known in modern IoT infrastructures. With a similar publish/subscribe model as the original Particle Cloud Server, MQTT can communicate between IoT endpoint devices without a need to communicate to a cloud system. With the ability to control the speed of the MQTT server ourselves, we can ensure the data is stored within the local network and is interpreted properly.

Numerous public MQTT server are available on the Internet. Their convenience gives us a benefit to prototype our system. However, its unpredictable traffic makes our system unstable and not always real-time. In our experience, we usually got a range of 30-second to 2-minute delays from sensor to subscription because of other users using the network. While packets on this network are reliable due to the MQTT framework being built upon the TCP protocol, this delay cannot be ignored.

To prevent those unstable factor from other users, we create a local MQTT server by our own. Connected with the wired network in our campus, this local server provides negligible latency. With our FitFeet sensors securely attached to our designed shoes and publishing to the local network, our system can organize the locations to compute its data according to its computing availability.

The FitFeet sensor can be attached to our designed shoes and publish the data it senses. Meanwhile, a powerful computer can subscribe this channel to get the instant data.

D. Data Packeting

Numerous packets are sent throughout the network in order to perform computation on available processors.

The format of the packet begins by gathering various data points on the FitFeet sensors. This raw data is then grouped together in one packet then is sent along the network to a more powerful computer for more complex computation. This computer then runs this data through the trained neural network previously mentioned and outputs the classification of the incoming results.

When the Machine Learning model provides its response, this classification data is then sent to the server for long-term provisions for our user. The data is then organized in terms of date and classification type per user for displaying purposes. Ultimately, without storing the raw data of the FitFeet sensors itself we solve the unnecessary need to recompute our data multiple times and store only the data needed on the server.

The final long term storage of our device only has the classification of the data and the time of the packet's arrival. This part of the system from the sensors, classifier, to the long term server reacts as the data arrives in order to prevent losses in data. This minimal data eventually stored on our long-term server allows us to display the statistics data needed for users on our external website.

E. SVM design

1) Preprocessing by normalization: Sending the raw data to the SVM naively results in a bad accuracy. The primary reason is the heterogeneity in each data dimension. For example, the value of accelerometer ranges from -50 to 50, but gyroscope is -500 to 500. This magnitude difference will still be preserved after FFT. Thus, a scalar is applied to make each dimension ranges from -1 to +1. Shown in figure 7, dashed line represents the value of each frequency component, which is divided by global maximum to make it fit into the plot. The solid line with normalization eliminates the bias caused from the unit of measurement or the sensitivity of the sensor.

2) Accuracy/Delay trade-off in SVM design: Accuracy and delay are two of the crucial components in the evaluation of this algorithm. However, there is a trade-off in this algorithm approach. As the previous section said, we designed an n-length circular buffer to store the incoming data. This length shows how long the signal is before, but on other side, it would involve a transient behavior change in response time. Shown in figure 8, when the user is walking (left figure), the circular buffer fills with the signal indicating a walking behavior. Once this user begins to run (right figure), the sensor generates a running signal and push it into the circular buffer. An ideal classification comes from a circular buffer filled with all the data corresponding to a run. However, this element-by-element overwriting spends some time replacing the whole buffer. This

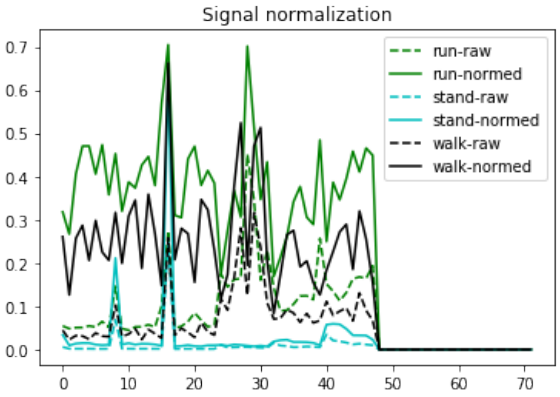


Fig. 7. Signal normalization

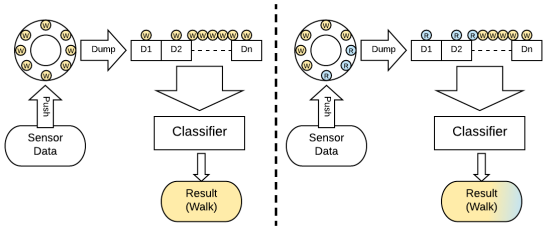


Fig. 8. Accuracy/Delay trade off

difference in time is considered the delay in response to the transient behavior change.

The responsive delay has an upper-bound. With the sampling frequency f_s, filling a n-length circular buffer takes $n*\frac{1}{f_s}$ seconds. In our configuration, f_s is 5 Hz, $n = 16$. In other words, we can interpret this as: when I begin to run from walk, I need 3.2 seconds for the classification to detect the change.

3) Accuracy versus window size: Intuitively, the larger window size gives more information and make the classification more accurate. In figure 9, we show the accuracy numbers by exploring different window size and training the model on each configuration. Each bar in the figure represents an average of 50 training sets to reduce the variance between each training. This results prove our hypothesis that larger window sizes have higher accuracy. Thus, given an application scenario (e.g. real-time, long-term), we can select a proper window size. In our case, we chose a window size of 16 to balance both computation time and accuracy.

4) Kernel selection: Using a kernel method in SVM gives a non-linear curve in high-dimensional space. For each point, we train our SVM model 10 times and then average the accuracy. This result shown in the following table gives us a reference of which kernel we need to pick for our application I. Here, the linear kernel performs the best out of the four kernels. We can also notice that the radial basis function (RBF) and sigmoid kernel also have good accuracy but not nearly enough accuracy as the linear model, and the polynomial kernel is the worst performing of our choices.

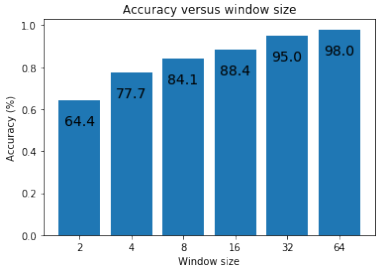


Fig. 9. Accuracy versus window size

	4	8	16	32
linear	76.38	82.68	88.79	94.03
rbf	74.80	81.34	86.73	89.73
poly	33.96	35.99	40.75	41.78
sigmoid	71.74	79.39	83.51	88.64

TABLE I
SVM KERNEL EXPLORATION

V. Conclusion and Future Plans

Overall, this project aims to create a prototype that combines pressure sensors and a 9-axis IMU to sense activity types, categorize them, and provide recommendations on bad mobility patterns. The results of our findings have proven that categorization of some mobility patterns such as running, walking, and standing can be done with great accuracy using our combined Machine Learning and FFT combined classification models.

The future steps in our design process is to make the device more physically appealing to our users for production. The pressure mapping system and analysis of the more complex behaviors can only be done after integrating a Pressure sensor network. This additional system can only be done with a solid design of the shoe itself because standard component are easily breakable and manual creation of an FSR network is very time-consuming to do precisely. Lastly, we aim to test our model on an entire system to ensure that each subsystem can be scaled to the customer market we envision.

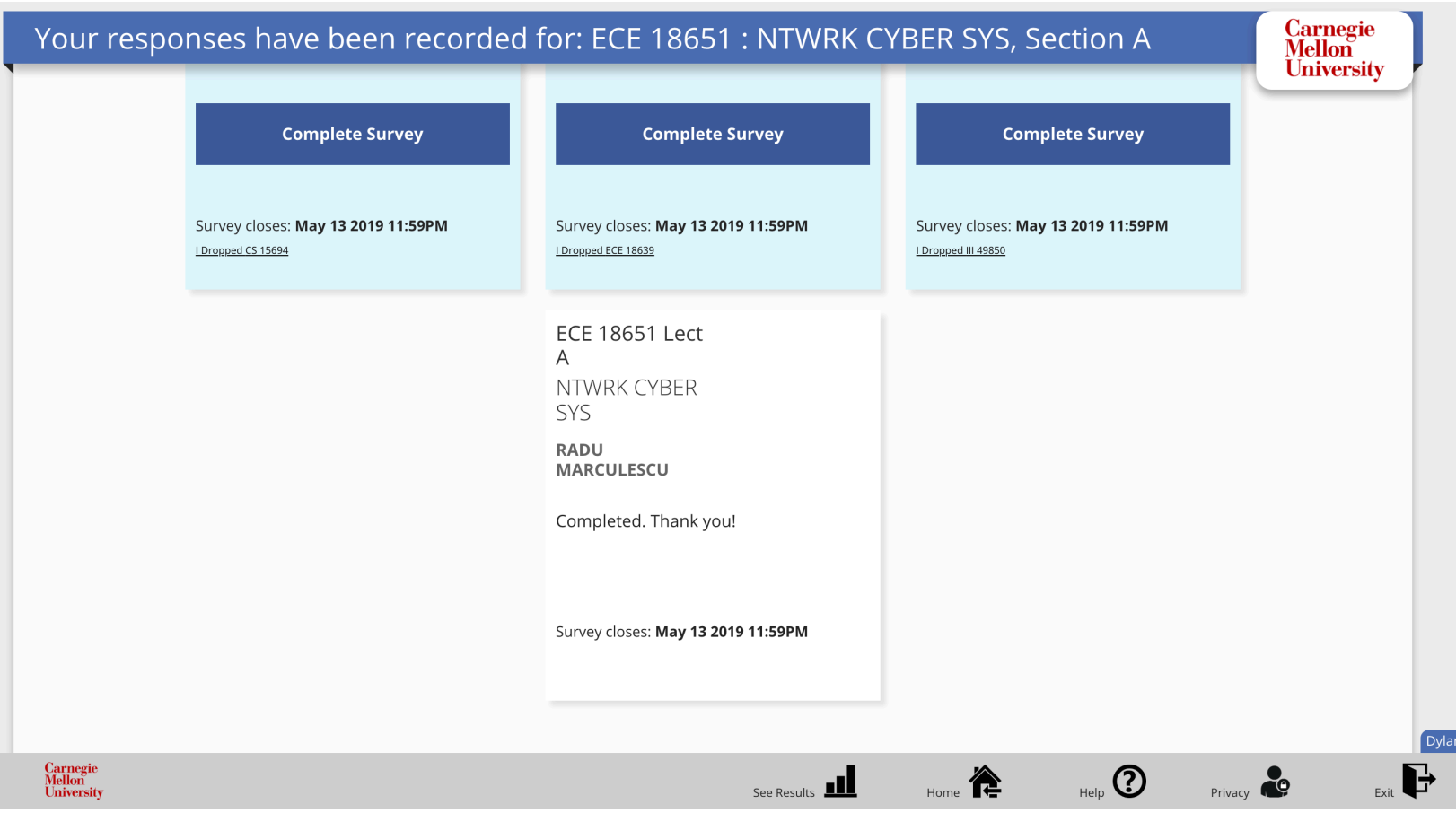
Contributions and questions that refer to the entire system interaction including computation locations, networking protocols, data visualization, and physical device construction, please contact author Dylan Vanmali.

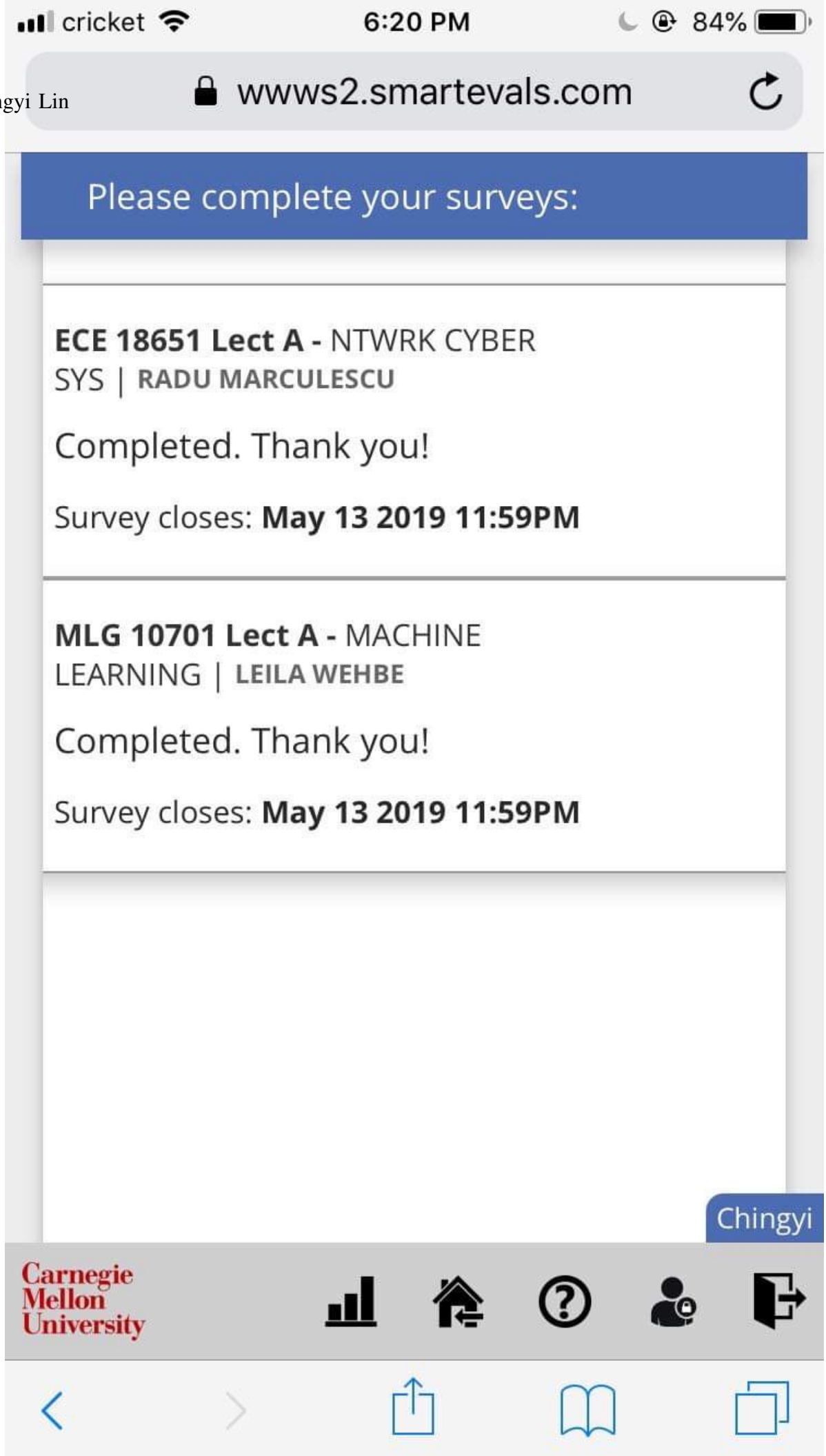
Contributions and questions towards the analysis and classification models for our system such as the FFT principles and the Machine Learning SVM, please contact author Chingyi Lin.

References

[1] https://en.wikipedia.org/wiki/Motion
[2] https://www.nanalyze.com/2019/02/smart-shoes-digitally-connected/?fbclid=IwAR3ZP5NlQGIsWpjKg4WdVavqlfdGuJhWZIRzeTZWYcSu4mHC6nr9AkkNODA
[3] https://create.arduino.cc/projecthub/380/smartinsoles-a42e49?ref=tag&ref_id=wearables&offset=46
[4] https://create.arduino.cc/projecthub/Juliette/a-diy-smart-insole-to-check-your-pressure-distribution-a5ceae?ref=tag&ref_id=wearables&offset=10

Dylan Vanmali




cricket
6:20 PM
84%
wwws2.smartevals.com
Chingyi Lin
Please complete your surveys:
ECE 18651 Lect A - NTWRK CYBER SYS | RADU MARCULESCU
Completed. Thank you!
Survey closes: May 13 2019 11:59PM
MLG 10701 Lect A - MACHINE LEARNING | LEILA WEHBE
Completed. Thank you!
Survey closes: May 13 2019 11:59PM
Chingyi
Carnegie Mellon University