Indoor Localization Framework with WiFi Fingerprinting

Project Member	Rajan Khullar
Project Advisor	Dr. Ziqian Dong
Semester	Fall 2016

Introduction

Abstract

As with most studies, the results of WiFi fingerprinting are much more meaningful if there is a large sample size. For this project I created an Android app that helps researchers in this field build their dataset efficiently. After collecting two weeks worth of data I studied how many access points are ideal to predict a user's location from a single WiFi scan.

Background

Global Positioning Systems (GPS) have been the standard technology used to obtain location of electronic devices. GPS devices are essential for the functionality of many of today's devices, appearing in vehicles, drones, and smart phones. They have a wide range of applications, from helping people navigate the roads to assisting the military in navigating planes and drones. Moreover, the popularity of smartphones with their inbuilt GPS devices has made GPS easily available.

Unfortunately localization using GPS is ineffective indoors and in highly metropolitan environments because of GPS signal fading. Signal fading occurs when signals penetrate building materials causing a decrease in intensity. This causes a decrease in signal-to-noise ratio making it difficult for GPS devices to differentiate between signals and noise. Signal fading is also a result of multipath phenomenon, which is caused by reflection and refraction of signals when they encounter walls [1]. Because of these reasons GPS signal is often lost entirely on smartphones in indoor environments, causing a pressing need for indoor localization.
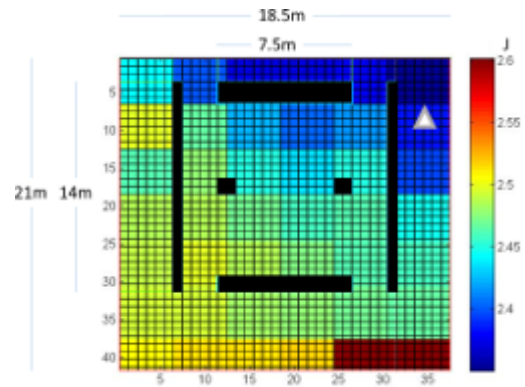
Indoor Localization may be utilized similarly to outdoor localization. This includes commercial applications such as real-time maps to help people navigate within malls or museums, and more technical applications such as guiding drones through indoor environments or locating cars inside tunnels. An indoor positioning system will open a new market of applications.

The most accurate way to express WiFi signal strength is with dBm, which stands for decibels relative to one milliwatt. Another common meaurement is the Received Signal Strength Indicator (RSSI). The RSSI measures the relative quality of the signal and is expressed with dB. The Android API provides signal strength as the latter. [5]

$$\mathrm{dBm} = 10 \times \log \frac{P}{1 \ \ \mathrm{mW}}$$	$$\mathrm{dB} = \log \frac{P_1}{P_2}$$
Absolute Power	*Ratio of Powers*
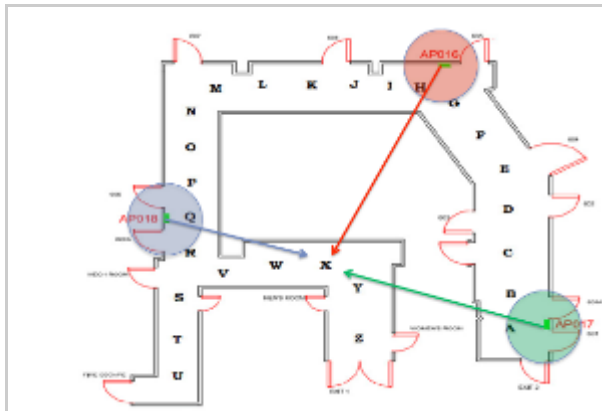
| Figure 1.1 - EGGC 6th Floor	Figure 1.2 - MC16 Auditorium

Related Work

Indoor Localization has been attempted through the use of Wi-Fi signal strength and Sensor fusion. Kothari et al (2012) [2] have successfully used dead reckoning and Wi-Fi signal strength fingerprinting to find the location of a smartphone. Dead reckoning was their method of using the accelerometer, gyroscope, compass and a particle filter in order to track walking and thereby track location. Both of these methods are prone to large errors. Wi-Fi signal strength is affected by obstacles and a myriad of other Wi-Fi signals in an urban environment, while the accelerometer and gyroscope sensors are likely to generate random noise in the data.

Thanks to the National Science Foundation's Research Experience Undergraduate (REU) program, research fellows have been able to study indoor localization at NYIT. As shown in Figure 1.1, students in the summer of 2013 chose three access points on a single floor and measured signal strengths from twenty six spots evenly distributed in the hallways. [3]

In Summer 2015 a student from Cooper Union and I were two of the REU fellows. We attempted to correlate energy consumption of a Nexus 5 to physical distance from an access point. We setup one router as a fixed access point and designed an Android application that records the current and voltage of the phone while pinging the router. We took ten samples at sixty points in a large classroom, however we were unable to find any significant correlation as shown in Figure 1.2.

In Summer 2016 two REU fellows studied multifloor localization with four consecutive floors in NYIT's main building. They were able to distinguish between thirty locations in their dataset with high accuracy. In all three REU studies the process of data collection proved difficult. The third project had a sample size of around 300 WiFi scans. After realizing this I was inspired to create a framework to help automate the process of gathering samples.

Implementation

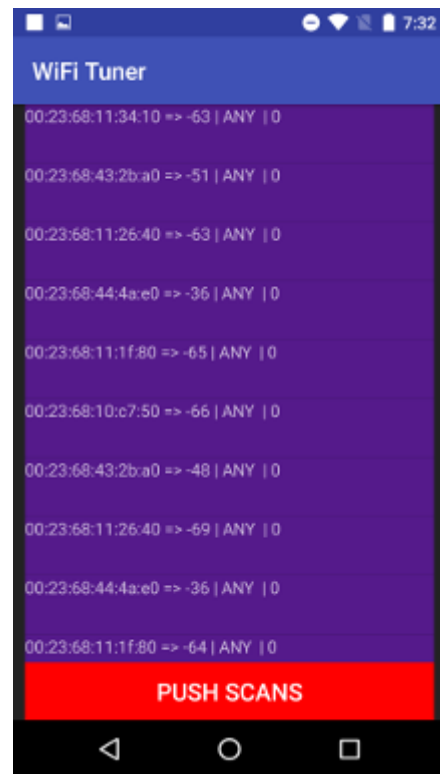***Figure 2.1 - High Level System Design***
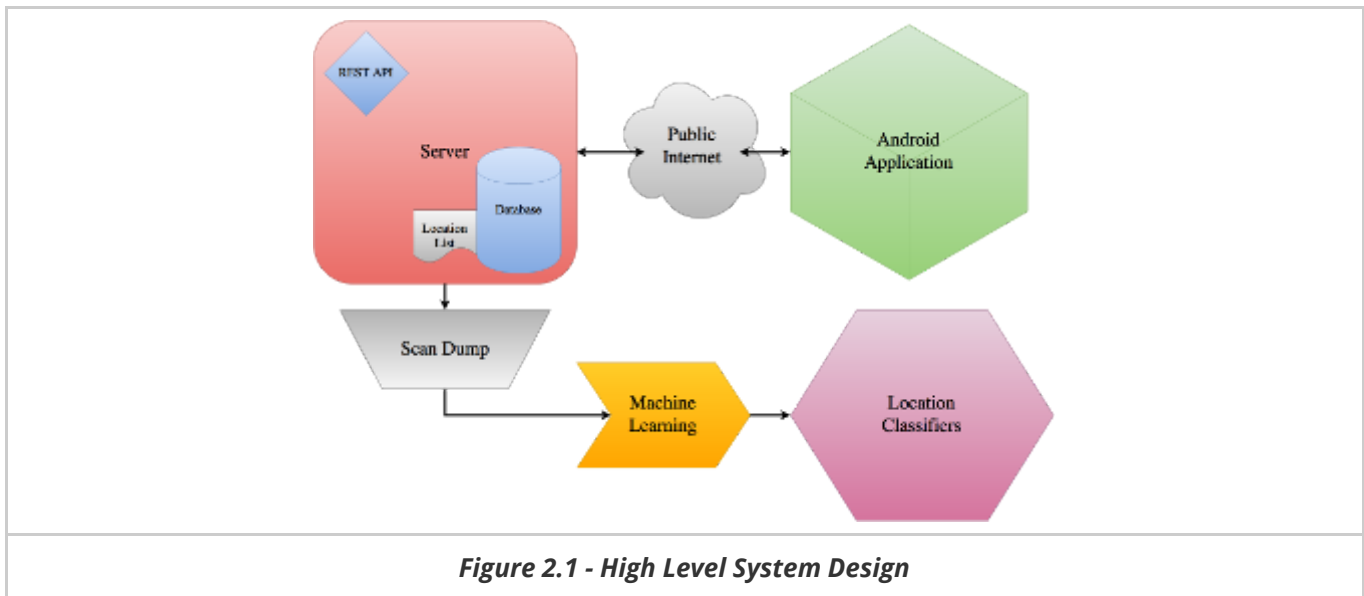
The Digital Ocean server has Apache and PostgreSQL installed. A python library called Flask was used to create a REST api. Java was used to create the Android application. As shown in Figure 3.1, once users sign up and login they can choose their classroom and setup a scan for the duration of that class. The scans will occur in the background so the users can close the app and use their phone normally. The scans can be paused or canceled in case the users needs to change their location. Finally once the scans are complete, then each user can upload their local dataset to the server.

For my analysis, the table of scan records if first downloaded from my server. Then it is preprocessed before being used to train the location classifiers. Before testing the prediction accuracy of the classification models, we analyze how much data we have in terms of both scan records and complete scans. We are also able to visualize the WiFi fingerprints for each location.

The graphs shown in this report represent only a small portion of all the graphs generated. All the graphs are stored on github with the following URL: rkhullar.github.io/csci870/
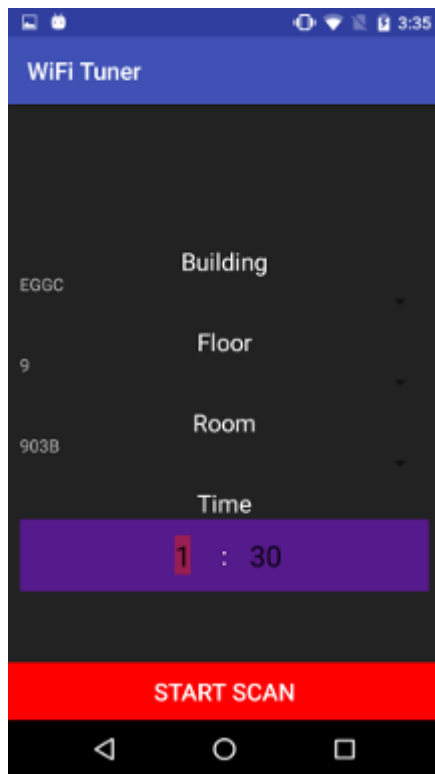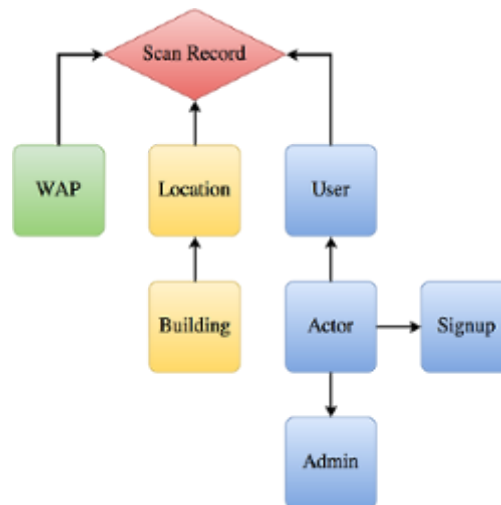
| Figure 3.1 - Setup Scan | Figure 3.2 - Push Scan Results |

Database

Figure 2.2 - Entity Relation Diagram

One WiFi scan or sample results in one or more scan records. Each scan record contains the the the mac address for a single access point, the signal strength to that access point, the location of the scan, the UNIX time stamp, and the user who performed the scan. The locations are stored as a separate table with building, floor, and room description.

In order to reduce redundancy a table of unique access points is maintained as well as one for unique locations. The actor table contains information for all people in the system including normal users, administrators, and new unverified signups. Figure 2.2 shows the simplified entity relation diagram.
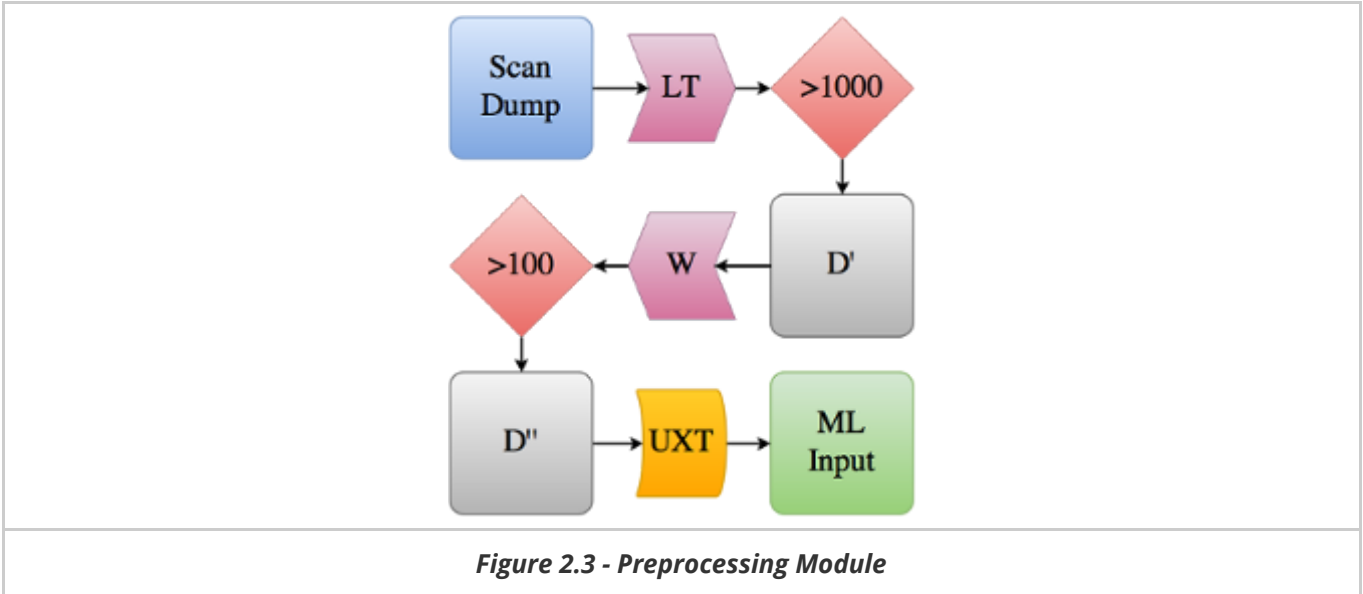
Preprocessing

Figure 2.3 - Preprocessing Module

Figure 2.3 shows how the dataset is prepared for machine learning. First the table of scan records is downloaded from the server by an administrator. The program groups each record by location and hour. Groups that do not have at least 1000 records are ignored. Each passing group is further grouped by the WiFi access point into blocks. Each block must contain at least 100 records. Then all the passing access points become columns and the passing records are combined by their timestamps into complete scans. The day of week and hour are also extracted from each timestamp.
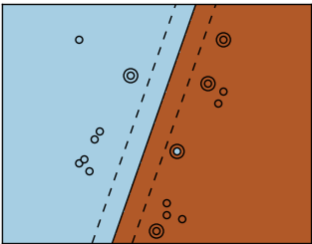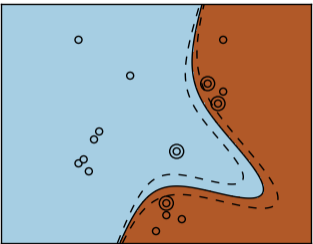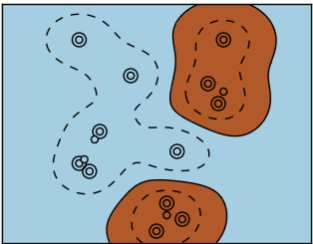
Input

UXT	BSSID	Signal Strength	Location
T1	W1	W	L1
T1	W2	X	L1
T2	W2	Y	L2
T2	W3	Z	L2

Output

DoW	Hour	W1	W2	W3	Location
dow(T1)	hour(T1)	W	X	-	L1
dow(T2)	hour(T2)	-	Y	Z	L2

Machine Learning

Figure 4.1 - Linear SVM	*Figure 4.2 - Polynomial SVM*	*Figure 4.3 - Radial SVM*

In this study we use the linear support vector machine classifier to perform location prediction. As shown in Figure 4.1 lines are generated to divide the hyperplane into labeled classes. Figure 4.3 shows an example of radial classification, but this usually leads to overfitting. Another countermeasure to this problem is to use k fold cross validation. The entire dataset of samples is randomly divided into k evenly sized blocks. Each block is used as testing data once while all the other blocks are used as training data. In this way k classifiers are actually trained and tested. [4]

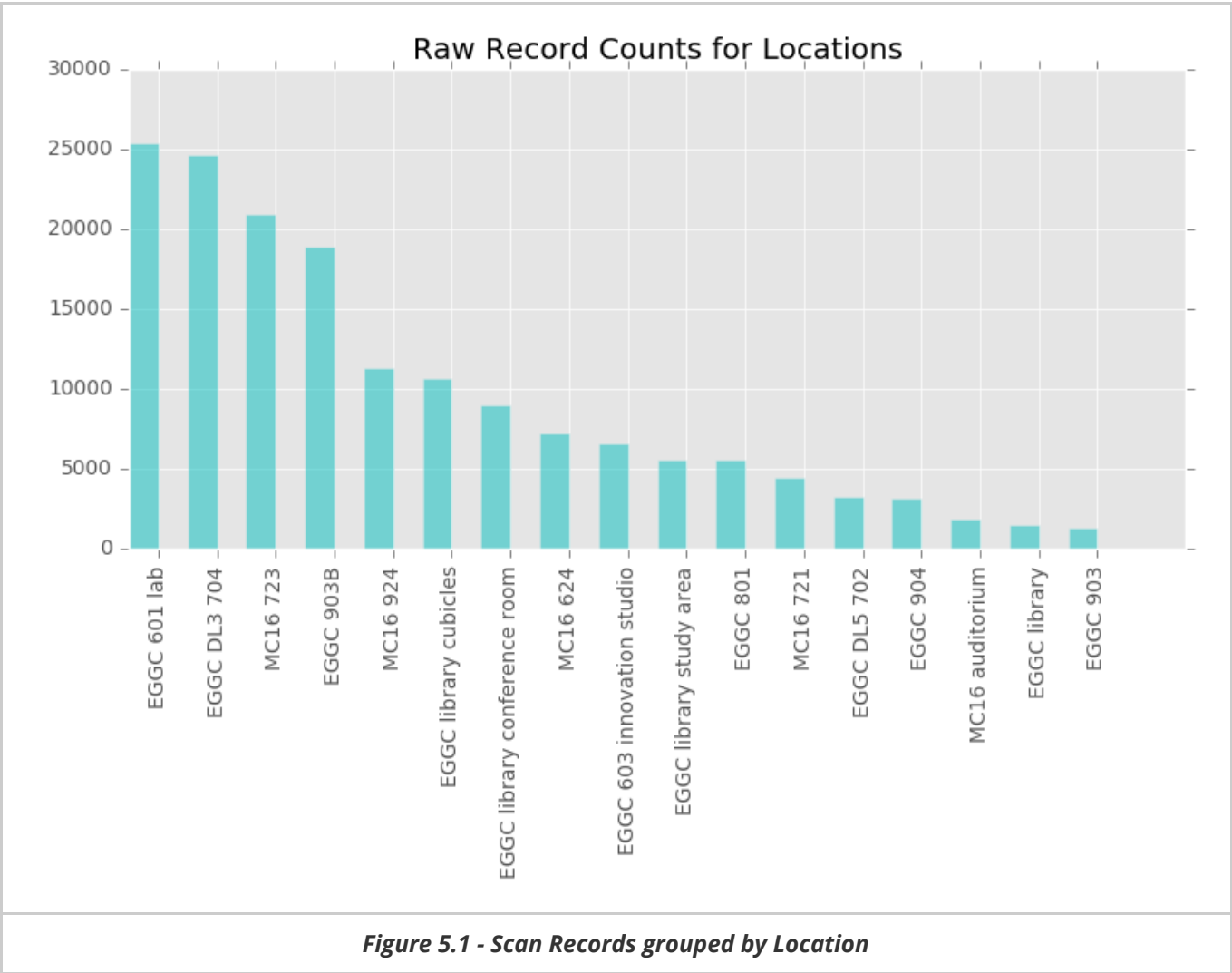# Results and Analysis

Group Cardinality

Figure 5.1 - Scan Records grouped by Location

Figure 5.1 shows that 17 locations passed the intial filter of having at least 1000 scan records. The top four locations have over 15,000 records.
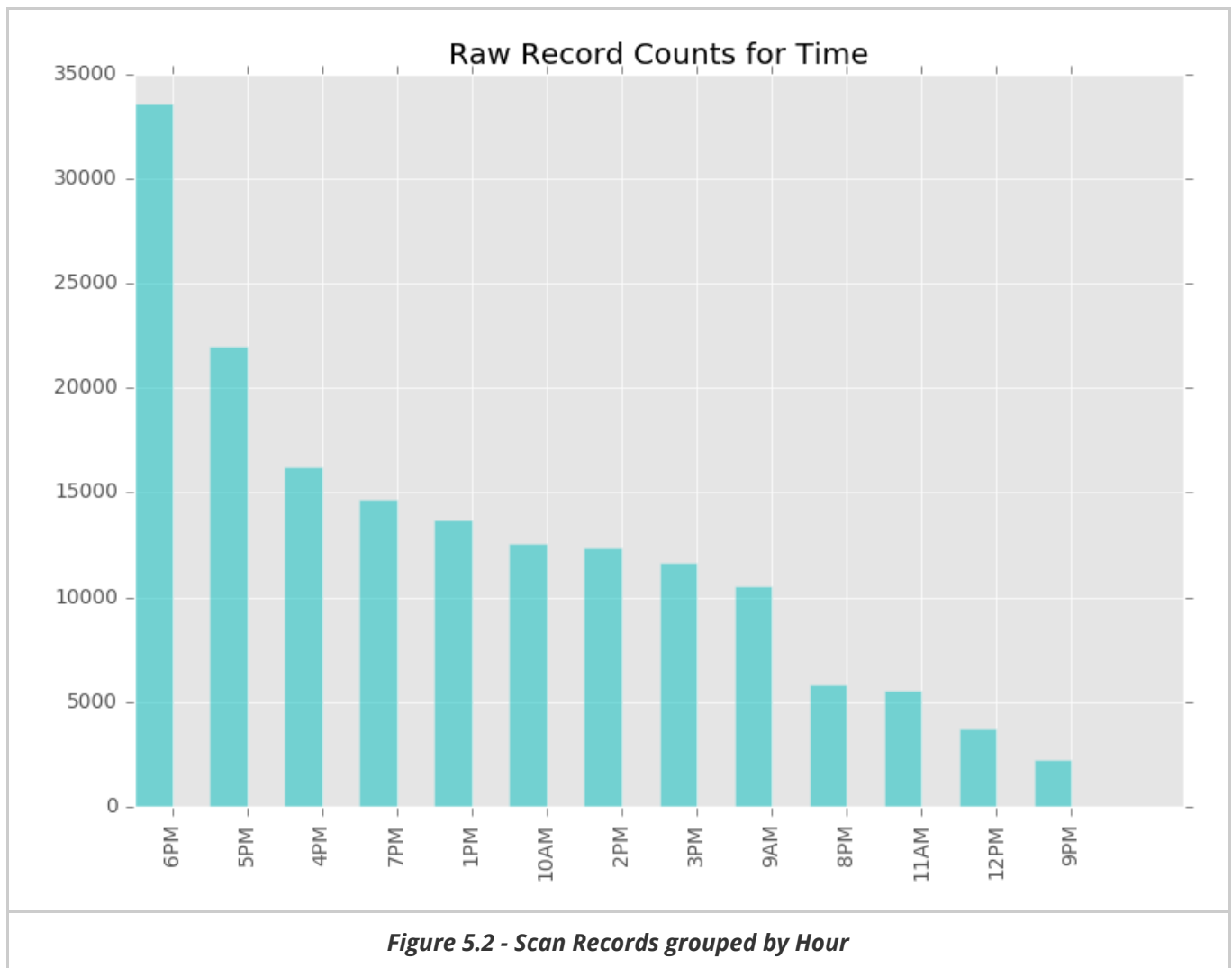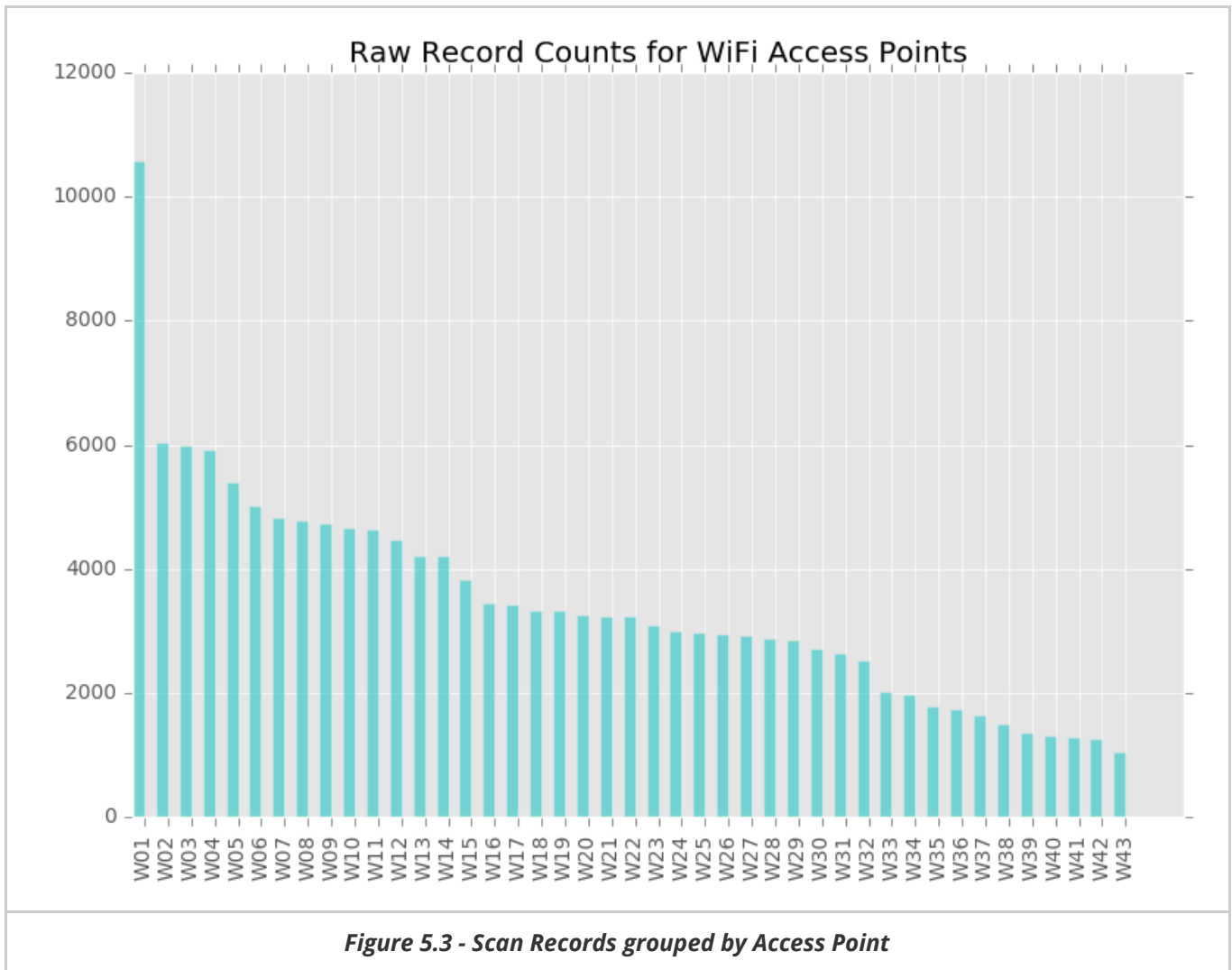
Figure 5.2 - Scan Records grouped by Hour

Figure 5.2 shows that most of the scans were performed between 4 PM and 7 PM.

Figure 5.3 - Scan Records grouped by Access Point

Forty three access points passed the intitial filter which means there are that many features plus two in total. The most common access point is nearly twice as prevalent as the second most common access point.

Figure 5.4 - Filtered Scans grouped by Location
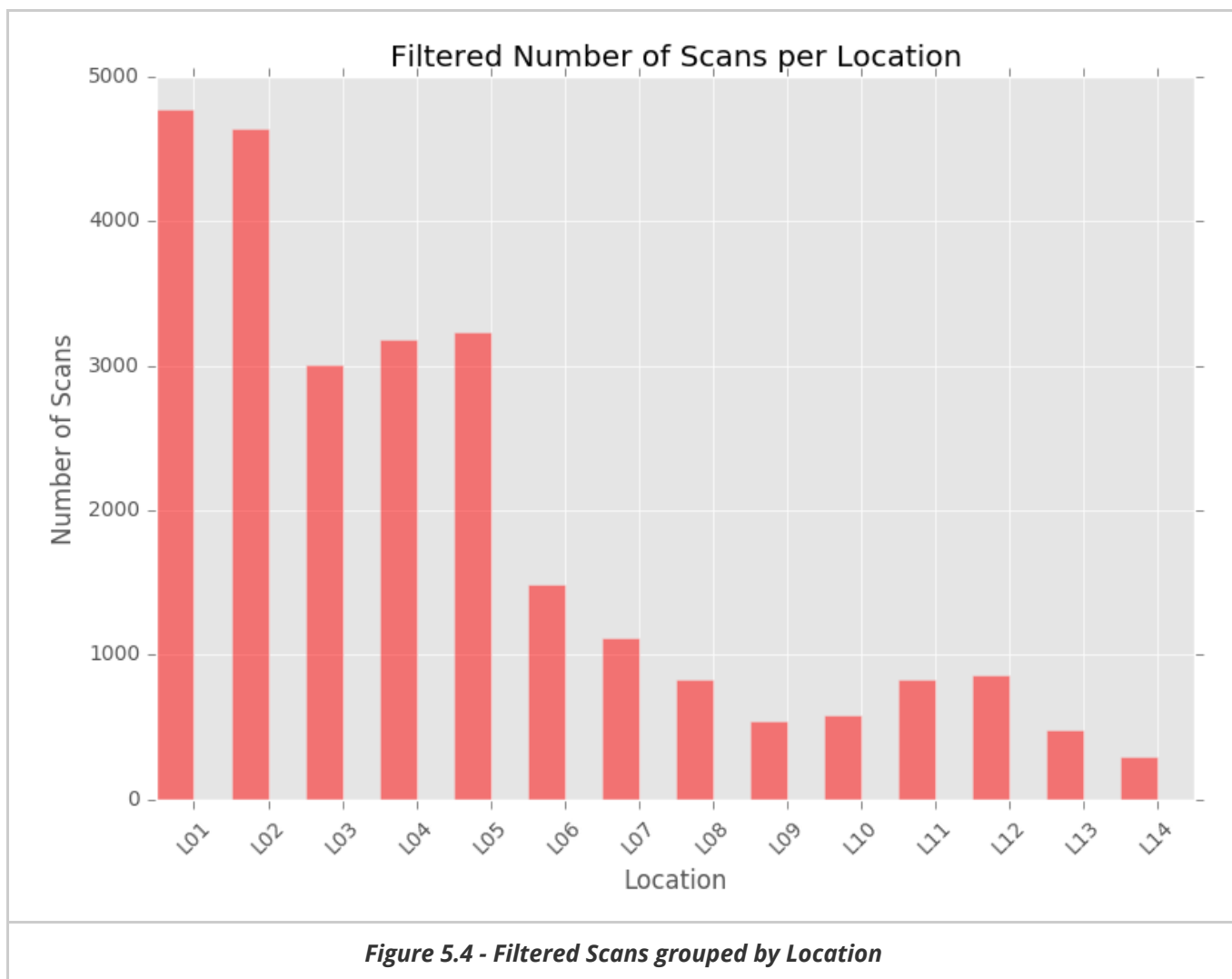
Before preprocessing there were 17 locations that passed the initial filter. After preprocessing only 14 locations are available to train the classifiers. Figure 5.4 shows how many scans there are for earch of those 14 locations. L01 through L05 have at least 3000 scans and the rest have under 1500 scans.

Signal Strength

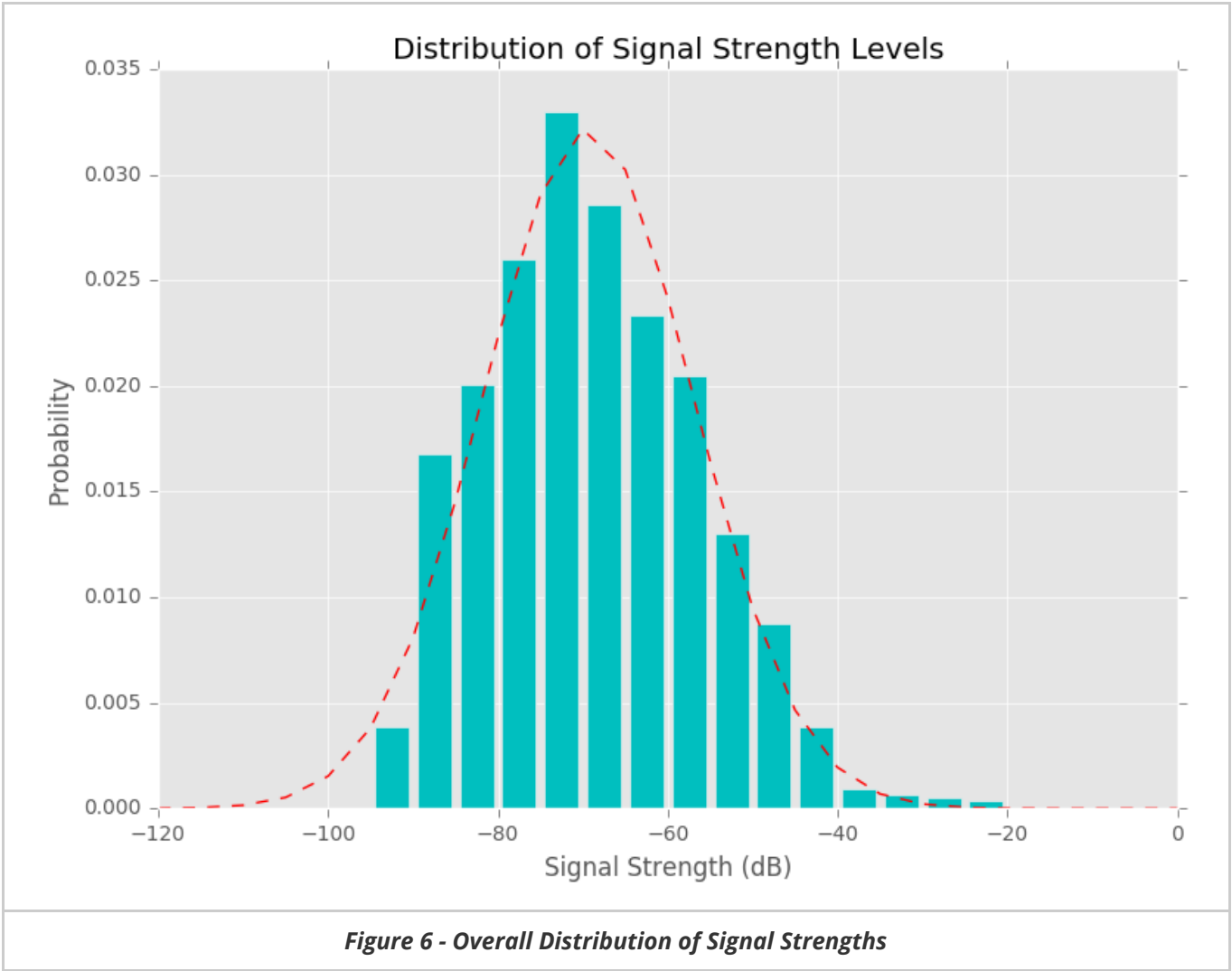
Figure 6 - Overall Distribution of Signal Strengths

Figure 6 shows that the best and worst signal strength's recorded in my dataset were -20 dB and -95 dB respectively. The signal strength is normally distributed with mean of -70 dB.
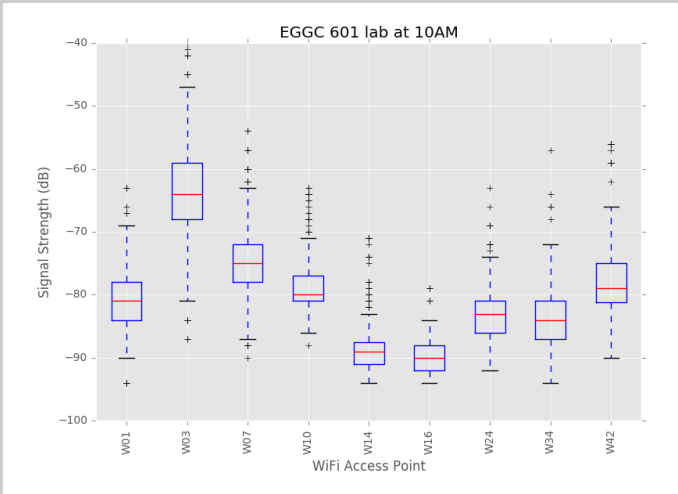
Fingerprints

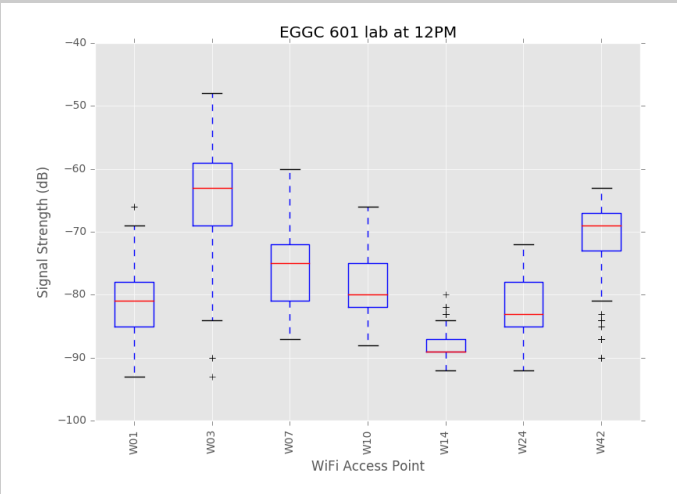Figure 7.1 - Fingerprint for EGGC 601 at 10 AM
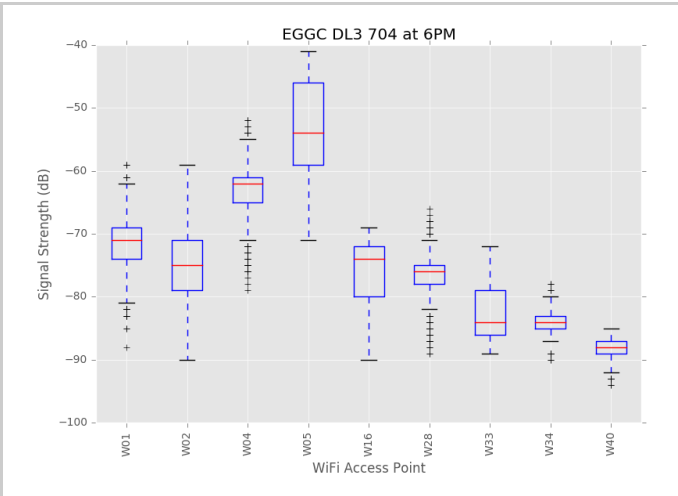

Figure 7.2 - Fingerprint for EGGC 601 at 12 PM
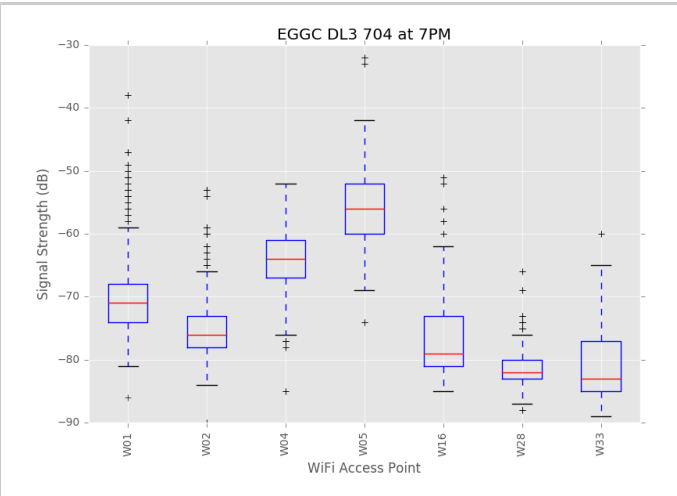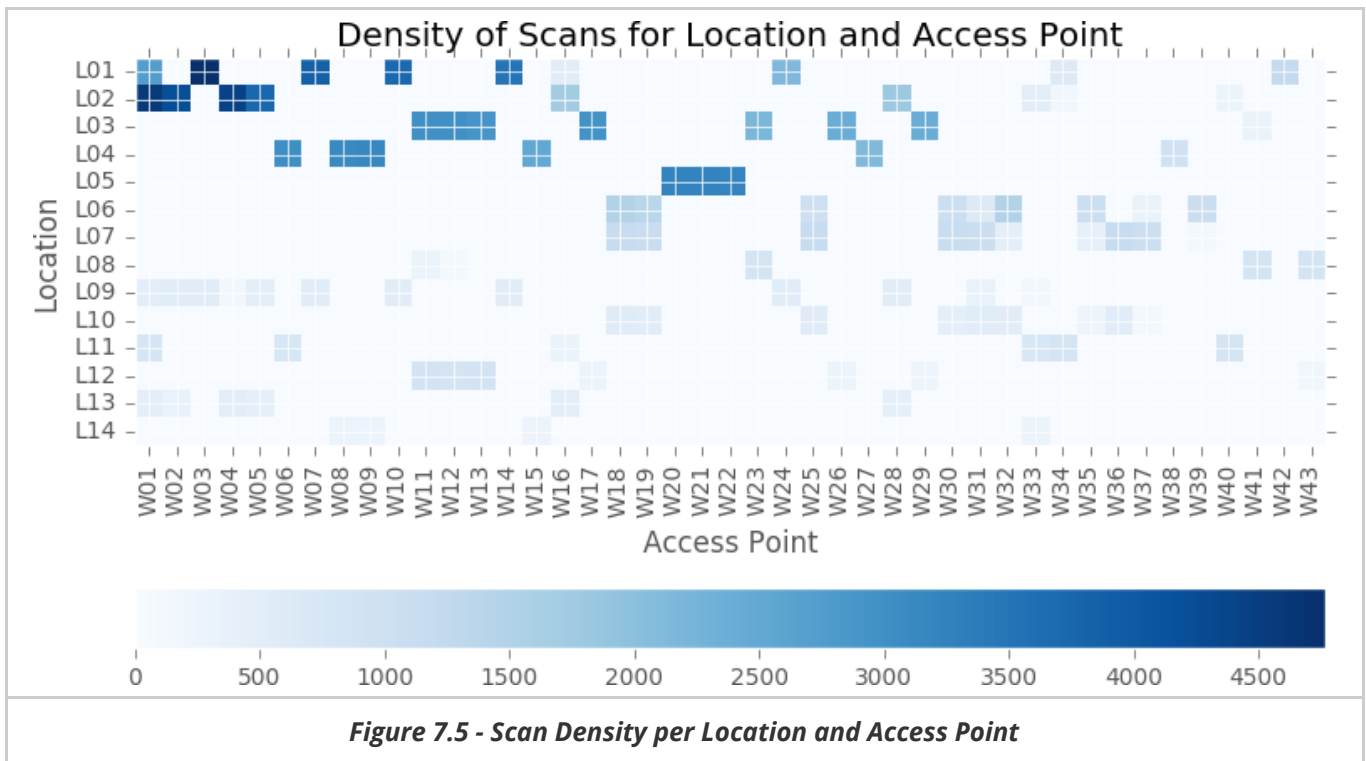

Figure 7.3 - Fingerprint for EGGC 704 at 6 PM

Figure 7.4 - Fingerprint for EGGC 704 at 7 PM

Figure 7.5 - Scan Density per Location and Access Point

Figures 7.1 through 7.4 show that WiFi scan fingerpints for individual locations are similar regardless of time, and that they are different for unique locations. Figure 7.5 shows how many samples are in each location and access point group. Each row can be interpreted as the fingerprint for one location as well.
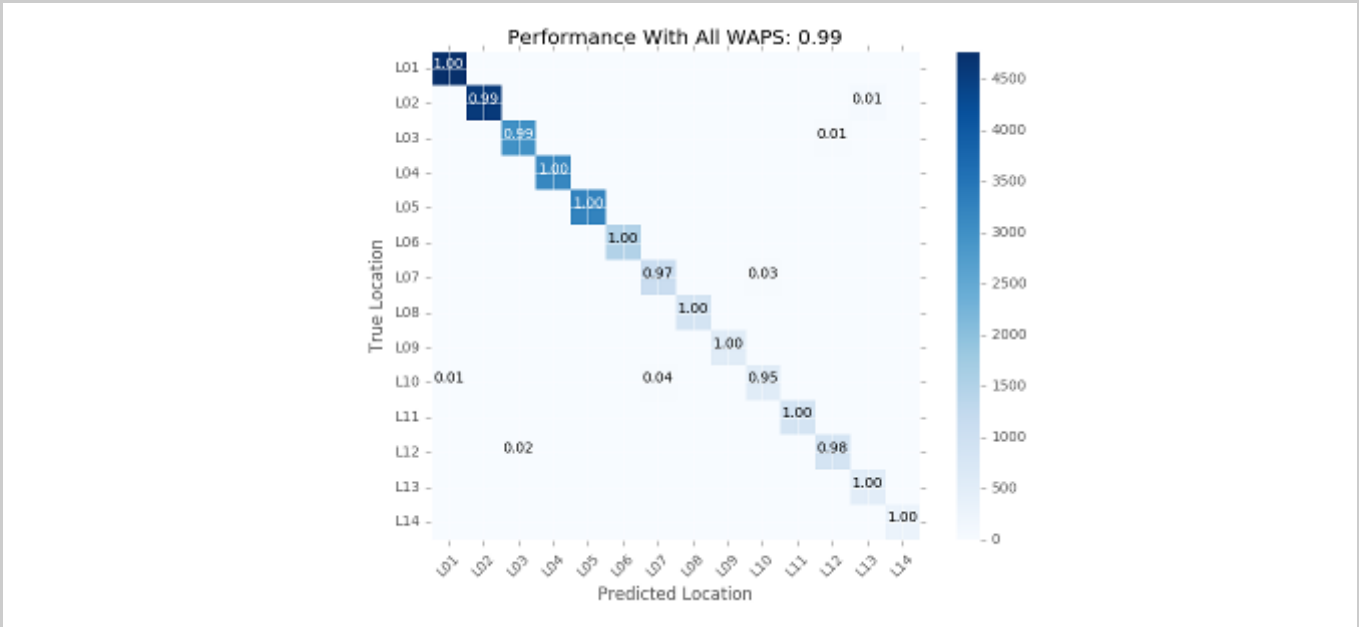
Prediction Accuracy

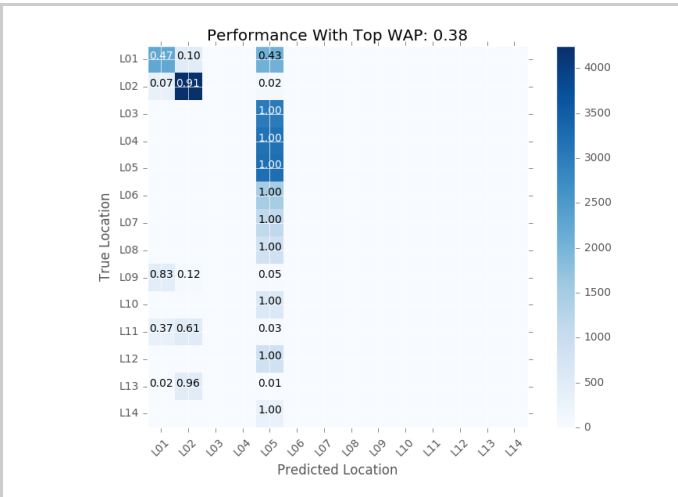Figure 8.1 - Performance with All Access Points
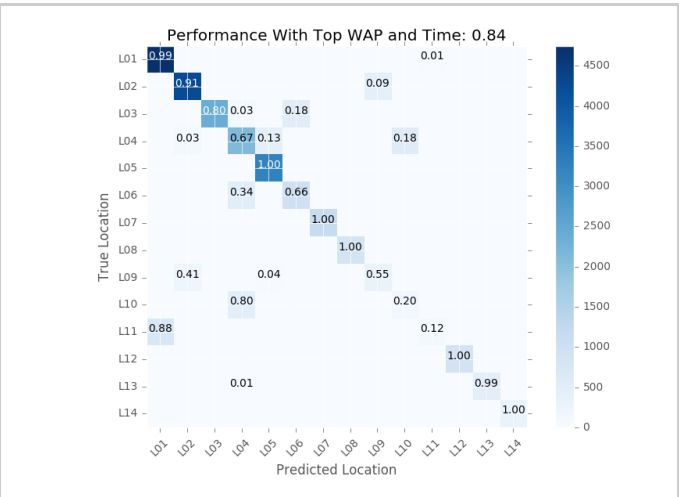

Figure 8.2 - Performance with One WAP

Figure 8.3 - Performance with One WAP and Time
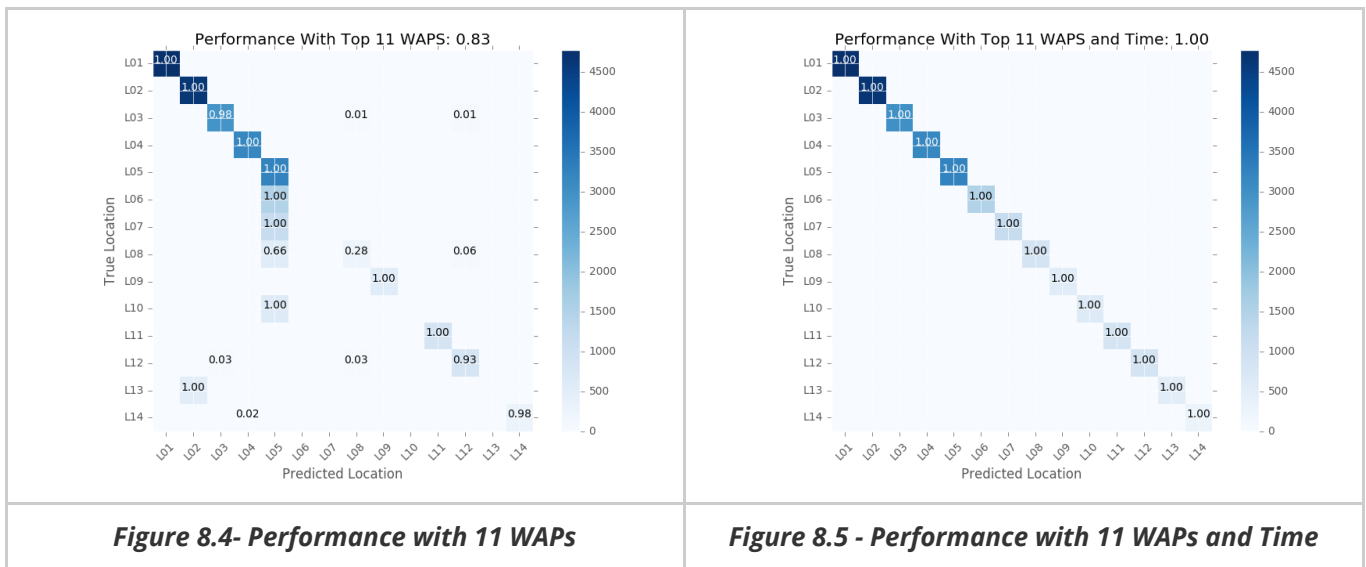
| Figure 8.4- Performance with 11 WAPs | Figure 8.5 - Performance with 11 WAPs and Time |

Figures 8.1 through 8.5 compare the location predictors that were trained from my dataset. Since 43 access points passed the inital filter there are a total of 45 features that can be used. Two of them are the day of week and hour of day, which are extracted from the timestamp for a scan. By using all the access points the location can be predicted with 99% accuracy. Most of the false predictions come from classifying scans from L10 as L07.

If only the most common access point is utilized then nearly all of the locations for each scan are predicted incorrectly as L05. The exceptions are samples from L02 and L05 which are still predicted correctly at least 90% of the time. By using the most common access point along with day of week and hour, the prediction accuracy significantly improves from 37% to 84%. By using the top eleven most common access points along with time, the prediction accuracy becomes 100%. Without time the accuracy is still less than when using time with only one access point.

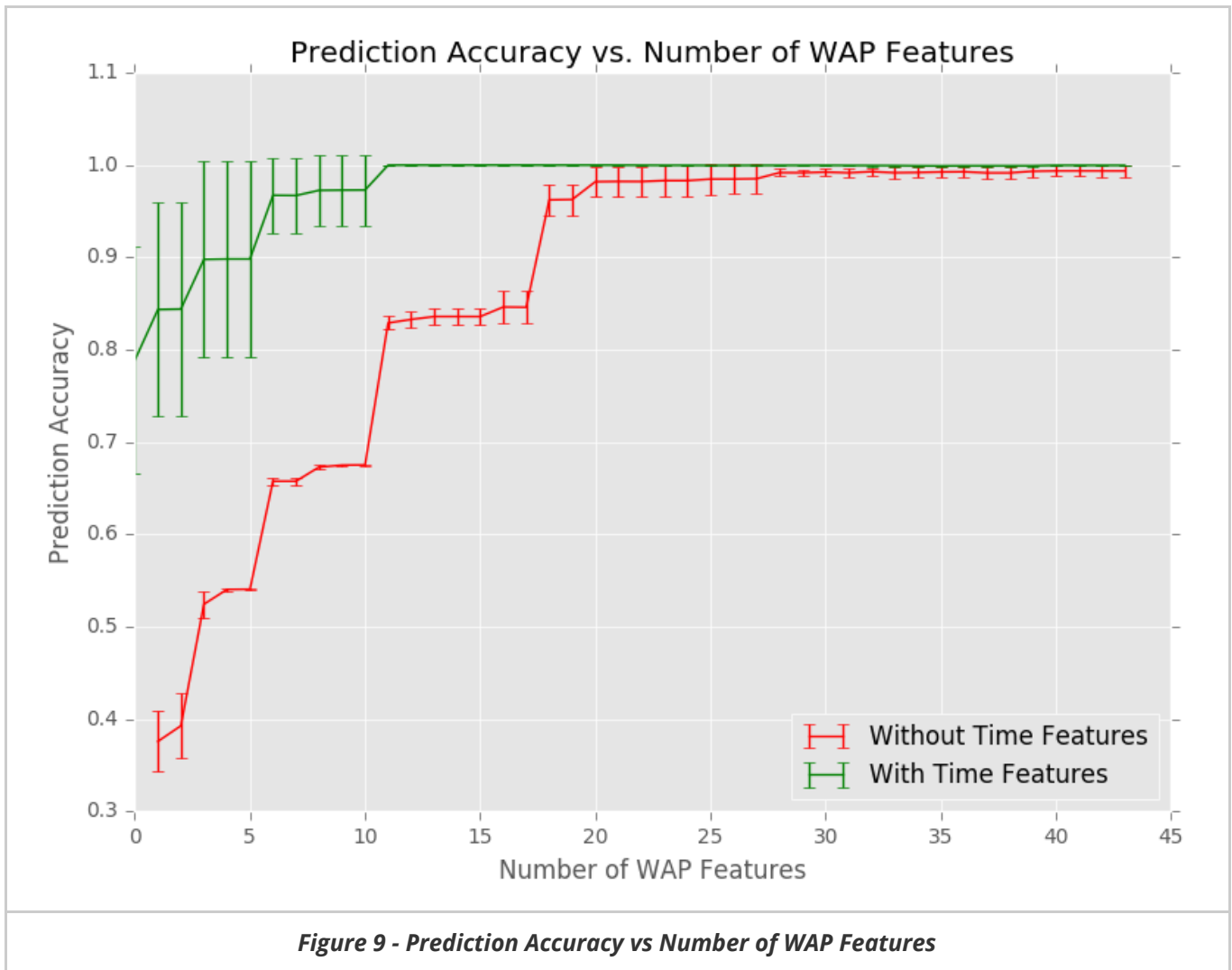*Figure 9 - Prediction Accuracy vs Number of WAP Features*

Figure 9 shows how the prediction accuracy changes with the number of access points used. Since forty three waps were used, there are eighty seven classifiers in total. The red line starts by using the most common access point. The green line start by only using the day of week and hour of day. Ideally using no access points should yield an accuracy of N^{-1} where N represents the number of locations to predict from. That would be the same as randomly choosing one location.

Since the classifier were trained and tested using 10 fold cross validation, there actually 10 accuracies for each one. The figure shows the average and and standard deviation for each classifier. The error bars on the green line are much larger then in the red line, which means that the scores for each individual classifier are much more consistent when only using the access points as features.

Conclusion

Prediction accuracy is significantly improved by including time features in the classification algorithm. With my dataset 100% accuracy can be achieved by using time and 11 access points. Without time the 28 most common access points are required to get close to the same accuracy. However as shown in Figures 5.1 and Figure 5.2, my dataset is highly unbalanced with regard to both time and location. That might be the reason we see such high improvement by using time as a feature.

Future Work

The Android application should be modified to record the phone's model number. This is important since the signal reading is dependant on the antenna and each model of phone may have a unique type of antenna. For example if an iphone and a nexus were to perform a WiFi scan at the exact same time and location, the results may not be the same. They should pick up the same access points but the signal strengths to each access point might be offset.

In order to easily balance the dataset, raspberry pi's can be placed in the perimeter of each room and be programed to automatically perform WiFi scans at a set time interval. When creating the location classifiers, the scans from the pi's should be used as training data. The scans from the mobile devices can be used as testing data. It would be interesting to see how accurate the predictors would be in that situation.

The signal strengths picked up by each type of mobile device may have a fixed offset to those picked up by the pi's. If that is the case then the training data would not need to come from phones at all and using one kind of device would suffice. Before predicting location with a scan from a phone, the signal strengths received would simply be offset appropriately.

Learning Outcomes

Server	Android	Machine Learning
PostgreSQL	SQLite	Training Classifiers
REST API in Flask	Broadcast Receivers	Cross Validation
Apache with HTTPS	Background Services	Confusion Matrices
Sending Email	Making HTTP Requests	Matplotlib
	Notifications	

References

1. Kjærgaard et al. "Indoor Positioning Using GPS Revisited," *Proceedings of the 8th International Conference on Pervasive Computing*, Helsinki, Finland. Springer-Verlag, 2010. 38-56
2. N. Kothari, B. Kannan, E. Glasgwow, M. Dias, Robust Indoor Localization on a Commercial Smart Phone. *Procedia Computer Science*, 2012. 1114-1120
3. N. Gutierrez, C. Belmonte, J. Hanvey, R. Espejo, Z. Dong, Indoor localization for mobile devices. *Proceedings of the 11th IEEE International Conference on Networking, Sensing and Control*, Miami, FL, 2014. 173-178.
4. [Scikit-learn: Machine Learning in Python](), Pedregosa et al., JMLR 12, pp. 2825-2830, 2011.
5. DB vs. DBm. Editorial. *InTech Magazine*, 1 Nov. 2002. ISA.