Indoor Localization Through Machine Learning on WiFi Fingerprints

Rahul Malavalli
University of California, Los Angeles
Email: rahul.malavalli@gmail.com

Arjun
University of California, Los Angeles
Email: arjun372@ucla.edu

Nilesh Gupta
University of California, Los Angeles
Email: nileshcgupta@gmail.com

Abstract—**This project attempts to provide a reliably accurate indoor localization service to complement the outdoor GPS systems so widely prevalent today. Using WiFi based RSSI fingerprinting and machine learning, we build and test Bayesian models in at-home and public locations, with f-measures of 0.893 for 14 points of interest in a hallway and 0.994 with 6 rooms within a house. The current implementation differentiates itself from the other various existing ventures because of its independence from a uniquely designed and installed infrastructure - the program runs indoor positioning by sensing and analyzing only the ambient signals of a given location without relying on explicitly placed beacons, or other signal generating objects. After detecting the powers (RSSI) of the numerous radio signals (WiFi was used for most tests), we use machine learning to identify numerous checkpoints set by the user during training. Future work will involve better GUI design for indoor navigation and the possibility of using different learning methods.**

I. Introduction

Over the years, GPS navigation has continually improved, pinpointing us to spots anywhere on the globe within an incredible accuracy of a few meters; we can safely drive or walk around on almost any road and confidently reach our destinations with little to no confusion. We might even venture to say that we've now engineered the perfect navigation system. However, in its current state, our navigation system utterly fails once we reach the door to our destination. By analyzing surrounding wireless signal profiles unique to different locations and their environments, this paper attempts to gather relatively accurate localization data that can be used for effective indoor navigation. By doing so, the system can enable many existing industries to create new value for their customers by leading prospective buyers to desired products in superstores, allowing students running late to class to more easily navigate their confusing campuses, empowering tourists to find their way around attractions, and helping many others in a multitude of situations.

II. Theoretical Background

Due to their proliferation in modern technology and society, radio frequency (RF) signals have been the target of numerous studies, and their general behavior in idealistic cases has been understood. These properties have been exploited in technologies prominent today, ranging from localization by triangulation in Global Positioning Systems (GPS) to mobile communications through cellular networks. The indoor localization methods in this paper specifically rely on the sometimes unnatural WiFi power distributions in indoor environments, as explained in the following sections.

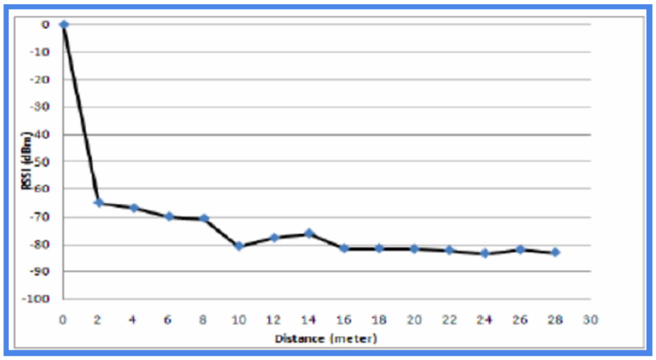

Fig. 1. RSSI (dBm) of RF Signal in Relation to Distance from Source

A. Relevant RF Signal Properties

It is known that all electromagnetic waves, and thus RF signals, decay in power by the inverse square law, as experimentally illustrated by Fig. 1 in an area with limited obstacles. When upheld, this property is crucial in localization methodologies involving triangulation.

However, all RF signals undergo multipath interference from uneven and unnatural obstacles in the real world, often resulting in not easily predictable power distributions. This effect is magnified indoors by the increased presence of doors, walls, windows, furniture, etc., that impede or otherwise manipulate a RF signal's power distribution. These interferences result in power distributions that don't follow a straightforward theoretical inverse power relation; instead, they often form unique plots not easily decipherable [1]. Instead of attempting to fit a general function manually to these unique distributions, such as is done in triangulation methods, this paper embraces the unexpected distributions in the methods described later on in the paper.

B. Signal Type Selection

As mentioned, the primary data source for this paper will be ambient RF signals. It was decided that out of the RF signals accessible to Android (WiFi 802.11x, Bluetooth, GPS, Cellular, and sometimes FM), WiFi would be the principal source of signal data for current system iteration. This was due to its balance of reliable, stationary node positioning (that act as ”beacons” to guide the system in indoor navigation) and its significant correlation of power with proximity (so that different distances from the source could be more easily distinguished). Other RF signals collected by Android, even if capable in staticity, perform poorly in RSSI variability, and vice versa.

C. Data Preprocessing Methods

The Android system returns a list of WiFi RSSI (Received Signal Strength Indicator) values with their respective timestamps periodically. Because every WiFi RSSI value is returned by the sensor with a unique timestamp on the microsecond time scale, the raw data results in an extremely sparse data set where only one valid RSSI is known per timestamp. Running 10-fold cross validation on this initially sparse matrix resulted in a very low f-measure of 0.520. It was hypothesized that increasing the density of the data matrix would improve predictive power by providing the classifier with more useful data to correlate locations with. To do so, data was ”binned” by time values ranging from microseconds to seconds, combining respective WiFi RSSI values into a single feature vector. Results of these new data sets are analyzed later in the paper. In the first iteration of the project, repetitions of RSSI values from the same WiFi node were resolved by simply taking the newest version; in the future, more sophisticated schemes can be used, such as weighted averages or fitting to Gaussian distributions, to generate more accurate approximations.

III. Implementation Overview

Much of the project is currently in the form of an Android application, and thus heavily relies on the Android code base for all of the graphical user interface and sensor access. Because Android runs on a Java Virtual Machine (JVM), however, all of the machine learning and data analysis was implemented in code relatively portable to other platforms (such as local use on computers and remote offloading onto servers) that do not use an Android-specific framework; rather, a popular third party program, Weka [2], was imported for use as the machine learning library of choice with this purpose in mind.

A. Training Pipeline

The general training and prediction pipeline follows the schematic in Fig. 2. As shown in the figure, the first step in the pipeline gathers data by making a ”first pass” for all immediate nodes and recording their current RSSI values. Following this, the data undergoes aforementioned preprocessing, which has been configured to automatically perform time-windowed binning. All feature vectors are then learned on using a basic Naive Bayes algorithm.

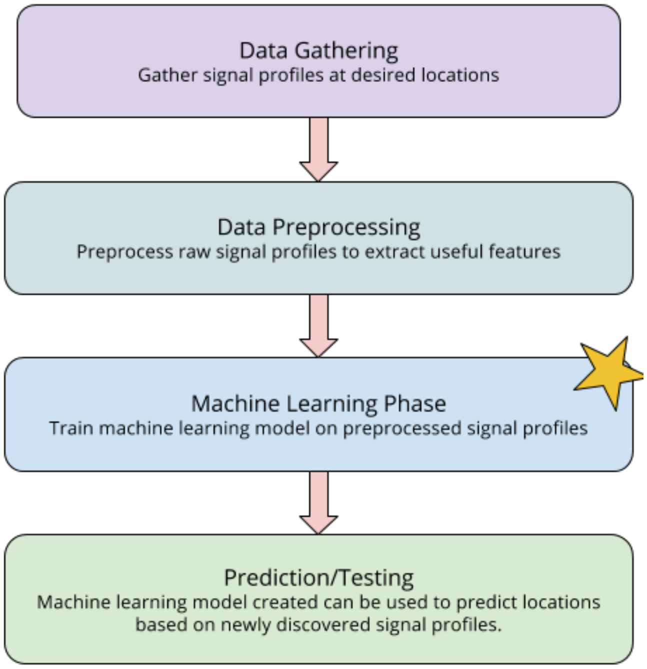
Data Gathering
Gather signal profiles at desired locations
Data Preprocessing
Preprocess raw signal profiles to extract useful features
Machine Learning Phase
Train machine learning model on preprocessed signal profiles
Prediction/Testing
Machine learning model created can be used to predict locations based on newly discovered signal profiles.

Fig. 2. Implementation pipeline of machine learning training and prediction.

B. General Application Structure

Four main functionalities were implemented on the Android side; the graphical user interface (cyan), set of motion and position sensors (green), wireless scanners (pink) and classifier (red). The other classes (blue) are helpers that make the code portable and scalable. The graphical user interface follows the general Activity - Fragment design principle with a central Activity that swaps and displays multiple Fragments that each contain different functional screens, such as training, predicting, etc. This allowed for wrapping some generally reused methods, such as indoor mapping access, in a single Activity that its respective Fragments could access. A visual representation of the Java-based class system can be seen in Fig. 3.

IV. Results and Other Evaluations

We evaluate our system in two general locations, one at home and another in a large hallway on a single floor of a university campus. In the home location, we identify six points of interest and collect data using the Android application, labeling and building a classifier in real-time. In the university hallway, we identify fourteen locations and follow the same steps, remaining within the hallway at all times. Fig. 4 illustrates classifier f-measures for both of these locations as a function of number of WiFi access points and binning width used in training. Since the hallway is larger and located on a university campus, 733 unique WiFi APs are detected whereas at home 111 APs are detected, each uniquely identified by BSSID. The number of WiFi access points were modified, in order of predictive power, to observe the effect different numbers of WiFi nodes have on predictions;

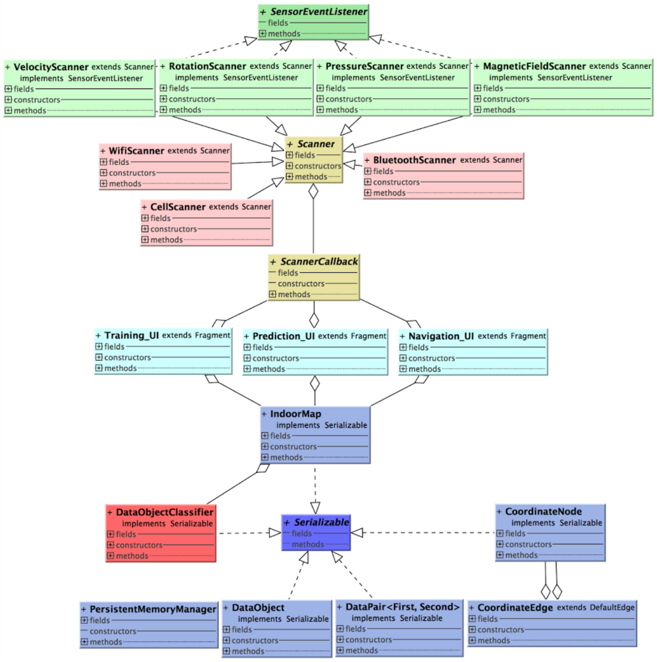
+ SensorEventListener
+ VelocityScanner extends Scanner implements SensorEventListener
+ RotationScanner extends Scanner implements SensorEventListener
+ PressureScanner extends Scanner implements SensorEventListener
+ MagneticFieldScanner extends Scanner implements SensorEventListener
+ WifiScanner extends Scanner
+ Scanner
+ BluetoothScanner extends Scanner
+ CellScanner extends Scanner
+ ScannerCallback
+ Training_UI extends Fragment
+ Prediction_UI extends Fragment
+ Navigation_UI extends Fragment
+ IndoorMap implements Serializable
+ DataObjectClassifier implements Serializable
+ Serializable
+ CoordinateNode implements Serializable
+ PersistentMemoryManager
+ DataObject implements Serializable
+ DataPair<First, Second> implements Serializable
+ CoordinateEdge extends DefaultEdge
fields
constructors
methods

Fig. 3. General Java-based class structure of the Android application.

this modifier was included to determine a number or ratio of WiFi nodes needed to obtain an optimal f-measure for easy profiling of a new environment. Generally, a positive correlation seems to exist between the number of WiFi nodes trained on and the f-measure from classifier. However, since only two different environments were tested in this paper, no significant conclusions could be made without more data from other environments.

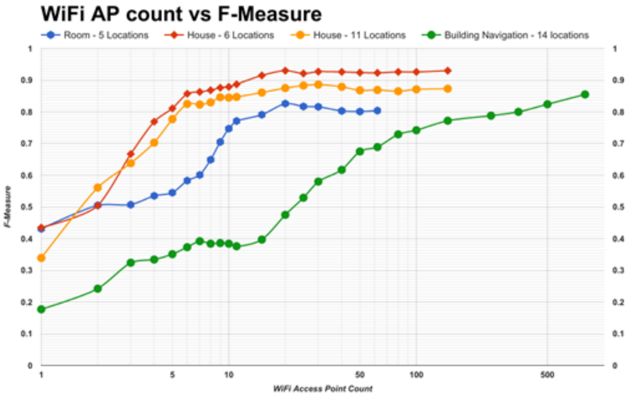

Fig. 4. Weighted F-Measure vs. unique number of WiFi APs

In addition to the total number of WiFi APs used in training, bin-width also affects the accuracy of the classifier. The two figures Fig. 5 and Fig. 6 illustrate f-measure on a heat map as a function of bin-size and number of WiFi APs discovered, for both the home and the university hallway environments. The X-axis depicts the bin-size whereas the Y-axis depicts the WiFi nodes. F-measures ranges are color-mapped to an eight color palette. Overall, it can be seen that no binning is much worse than even slight amount of binning. However, over-binning leads to increased error since the user may move away from the location quickly depending on the distance between adjacent locations. In a hallway, adjacent locations may be traversed in less than 10 seconds but in a home this number may vary significantly. In the home in question, the average time spent in each location was 20 seconds. The accuracy of the classifier can therefore been seen reducing in multiples of 20 seconds starting from the 20 second time division. After the 40 second time interval, the classifier accuracy for home has dropped below 0.60.

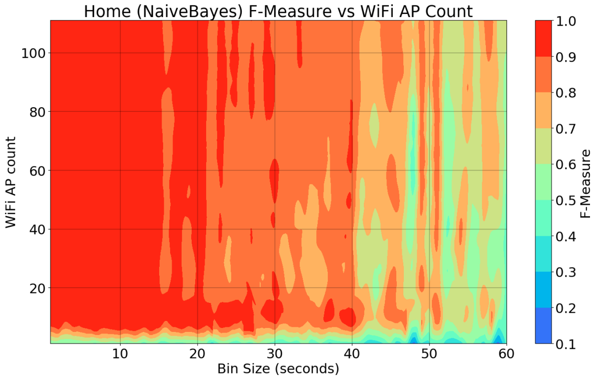

Fig. 5. Heat map, with 6 classes in a home environment, of f-measure in relation to number of WiFi nodes and amount of binning (in microseconds).

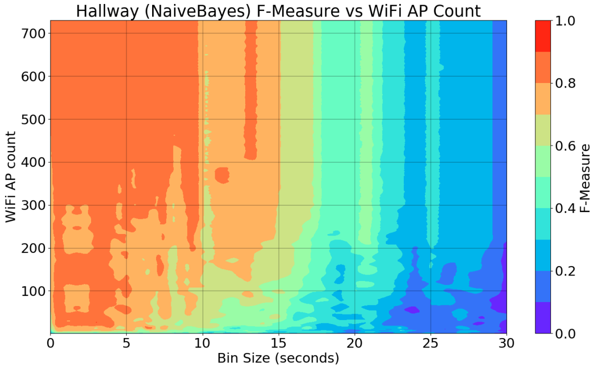

Fig. 6. Heat map, with 14 classes in a university building environment, of f-measure in relation to number of WiFi nodes and amount of binning (in microseconds).

A. Comparisons to Other Methodologies

We now offer a brief comparison of other popular localization methodologies. To sharpen the focus of our paper, we restrict the comparisons to technologies also utilizing WLAN signal data. This eliminates the need to look at positioning solutions with lower accuracies such as Cellular-based (50 - 200 m) and GPS-based (5 - 50 m) localization technologies, and narrows our domain to exclude methods requiring additional infrastructure (for example, Bluetooth beacons) [3]. The first of these is the RADAR tracking system proposed by Bahs et al [4] which obtains the users current position by looking at the signal strengths of k nearest neighbors. This signal-space technique is dependent on user orientations, amount of noise, and the number of neighbors used, all of which can be highly sensitive to differences in building materials and other environmental factors. Due to our projects utilization of all readable nodes, we can avoid some of this variance; specifically, using all of the readable nodes in a setting means we lessen the effect of device orientation and reduce the amount of interference through the use of learning algorithms that do not necessitate a strict fitting model. With the second iteration of the project, RADAR achieves roughly 50% accuracy from 2-3 m.

The horus system [5] is another WiFi-based positioning technology that offers a joint clustering approach to estimate user location. Each target coordinate is treated as a separate class possessing a certain probability. The system returns the candidate location for which the likelihood is highest; the project compared results in an approximately 90% accuracy at mean distances of 2.1 m. Similarly, another location estimation system uses a neural-network based classifier [6] to achieve accuracies of 72% at 1 m and up to 95% at 2 m. These fall in the range of our f-measure of 90% at 2-3 m using a bin size of 5 seconds with 580 access points, and even fall short of our best f-measure of 99.4% at 3-4 m with a bin size of 18 seconds and 8 access points. Thus, the machine learning and binning procedures utilized in our project represent promising techniques in localization and prompt further investigation.

V. Conclusion

The system first began exploring WiFi-based indoor positioning and navigation by learning from raw RSSI data at a surprisingly low f-measure of 0.520. However, this was before any preprocessing, and thus left vast room for improvement. Two main steps were taken to improve the model being used; determining a classifier best fit to the problem at hand, and basic preprocessing to reduce the sparsity of the dataset.

After experimenting with different classifiers, it became clear that the Naive Bayes classifier was the best equipped, in the environments chosen for this paper, to effectively learn and predict on this type of sparse data set. Then, binning of microsecond timestamps to multi-second long windows was employed to immensely increase data set density; further experimentation is necessary in this area to determine the most optimal binning time-window, as well as to explore dynamic binning based on the accuracy radius required, type of location environment, type of motion encountered, and more. From these steps, the accuracy in a home environment increased to 0.994 (with multimeter location radii) and around 0.893 in building classification at a 1-2 meter location radius.

This system can be improved in further iterations by exploring the benefits of data preprocessing outside merely the realm of simple time-based binning. More features can be generated to extract additional useful information from the raw data and reduce the time spent in training; statistical metrics, such as mean, standard deviation, or other Gaussian distribution statistics, can be derived in conjunction with motion data to coordinate dynamic binning intervals, among others. By continuing to improve the proposed system, we aim to bring the outdoors in, making indoor navigation as simple and easy to use as GPS is for the outdoors today.

References

[1] S. Shue & J. M. Conrad, Reducing the Effect of Signal Multipathing in RSSI-Distance Estimation using Kalman Filters, 19th Communications & Networking Symposium (CNS 2016), 2016.

[2] The University of Waikato, Weka3: Data Mining Software in Java, Weka 3 - Data Mining with Open Source Machine Learning Software in Java. [Online]. Available: http://www.cs.waikato.ac.nz/ml/weka/index.html. [Accessed: 14-May-2017].

[3] H. Liu, H. Darabi, P. Banerjee, & J. Liu, Survey of Wireless Indoor Positioning Techniques and Systems, IEEE Transactions On Systems, Man, and Cybernetics - Part C: Applications and Reviews, Vol. 37, No. 6, November 2007.

[4] P. Bahl and V. N. Padmanabhan, RADAR: An in-building RF-based user location and tracking system, in Proc. IEEE INFOCOM 2000, Mar., vol. 2, pp. 775784.

[5] M. Youssef, A. Agrawala, and A. Udaya Shankar, WLAN location determination via clustering and probability distributions, IEEE Int. Conf. Pervasive Comput. Commun., Mar. 2003, pp. 1431

[6] R. Battiti, T. L. Nhat, and A. Villani, Location-aware computing: A neural network model for determining location in wireless LANs, Tech. Rep. DIT-020083, 2002