WiFi Fingerprint Indoor Positioning System

Braxton Adams[1], Cyrus Cravens[2], Jonathan Adiri[3], Sang Xing[4]
Department of Mathematics and Statistics
Portland State University, Portland, OR 97201 USA
[1]braadams@pdx.edu, [2]ccravens@pdx.edu, [3]adiri@pdx.edu, [4]sxing@pdx.edu

***Abstract*—To be continued...**

***Index Terms*—WiFi fingerprinting, WiFi indoor positioning, Trilateration, K-nearest neighbor algorithm**

I. Introduction

In recent years, with the ubiquity of mobile and Internet of Things devices, the possibility of creating an Indoor Positioning System (IPS), has become a reality for many structures. Our client, the engineering department of a university, has requested our team to create a model that predicts the physical location of a device, based on a set of signal strengths between access points and connected devices. The client also asked our team to quantify the accuracy and precision of the model. In order to find the most accurate way of predicting the location, two approaches were used in this experiment, Trilateration and the K-nearest Neighbors algorithm (KNN) and The experimental results showa that our model has been able to predict the location of a connected device to an average positioning error of 8 feet.

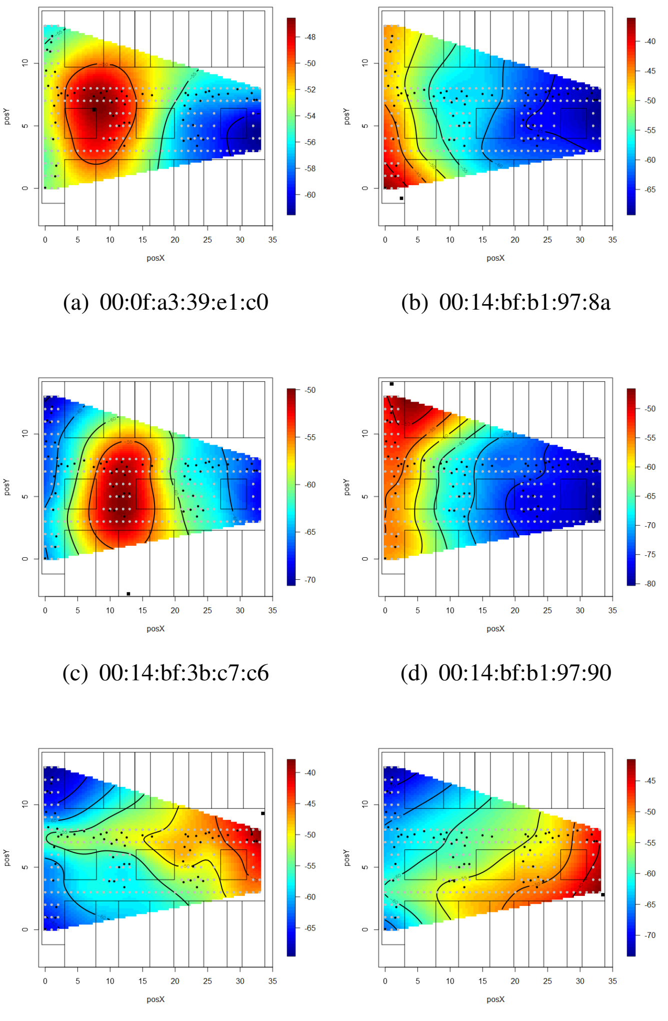
Our group was contacted by a local university who collected data from their internet network. The client requested that we identify the physical location of devices that are connected to the university network. There are six access points (i.e. wifi routers), with distinct MAC addresses, on a certain floor of one of the university buildings. The floor-plan is roughly 35mx18m and is composed of several class rooms and hallways. Devices that are connected to the network can measure the Received Signal Strength Indication (RSSI) of all access points within range. To collect the data, a mobile device connected to the network has recorded different values at various locations and oriented in various directions. The collected data was split into Offline and Online data, which will later be used to train and test the models developed by our team. Two approaches were tried. The first, Trilateration, is the use of distances for determining the coordinates of a point in space. Since this approach is being used for geopositioning, we thought that it might be applied to IPS as well. The second approach, KNN, is a non-parametric supervised learning method. This approach seems promising because it has been proven successful in the past at solving similar problems.

II. Data

III. Methodology

- Concise and intuitive description of methods considered
- Results
- Re-state the points of project and show how these were tackled and what resulted from analysis
- Include most relevant results to display, others go in appendix

There are many different statistical techniques we can use to estimate the location of a device from the detected RSSIs between the device and several access points. Another method we adopted is using the K-nearest Neighbors (KNN) Classification algorithm in order to predict the locations of our devices from the online dataset. The procedure of this method is as follows: we first have our entire dataset consisting of offline and online data, we these transform the offline data as our training dataset and online data as our testing dataset. Since the testing dataset has records of RSSI values from various locations, we take these RSSI values compare with the close neighbors (in terms of RSSI) from out training dataset. By close, we mean finding similar RSSI values from training dataset under the assumption that the predicted location has the same



(a) 00:0f:a3:39:e1:c0

(b) 00:14:bf:b1:97:8a

(c) 00:14:bf:3b:c7:c6

(d) 00:14:bf:b1:97:90

(e) 00:14:bf:b1:97:8d

(f) 00:14:bf:b1:97:81

Fig. 1. Heat map of 6 Access Points in all orientations

MAC address and orientation as the actual location of the testing dataset. We define k as the number of neighbors in the training set, then the corresponding positions of our neighbors will be considered as the predicted positions in the test dataset. The neighbors are defines to be

$$\min\left(\sqrt{(RSSI_{train} - RSSI_{test})^2}\right), k = 1.$$

If $k > 1$, then we choose least k numbers of RSSI values as our neighbors.

Now that we have the basic procedures as to how to find the predicted positions, this brings up a new question as to how many neighbors should we select? The choice of k is very sensitive, choosing a small amount of k leads to underfitting of our model to the training data, and choosing a large amount of k leads to overfitting. Ideally, we want to choose the value of k that is independent from our test data, the method of k-fold Cross-Validation (CV) can help us do this.

The idea behind k-fold CV is quite simple: we first divide our training data into numerous non-overlapping subsets of equal size. Then for each subset, we build models with the data that are not in that subset and we assess the predictive ability of the model using the subset that was left out. We repeat this model fitting and assessment for each folds and aggregate the prediction errors across the folds. The prediction errors are denoted as

$$\text{SSE}(XY_{act} - XY_{est}),$$

where SSE stands for Sum of Sqaures Error or Sum of Sqaures Residuals, representing the difference between the actual postilion and the estimated position of devices (see Fig 2).

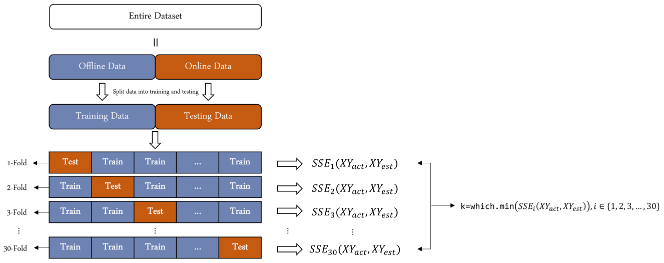

Entire Dataset
||
Offline Data
Online Data
Split data into training and testing
Training Data
Testing Data
1-Fold
2-Fold
3-Fold
30-Fold
Test
Train
...
$SSE_1(XY_{act}, XY_{est})$
$SSE_2(XY_{act}, XY_{est})$
$SSE_3(XY_{act}, XY_{est})$
$SSE_{30}(XY_{act}, XY_{est})$
k=which.min$(SSE_i(XY_{act}, XY_{est})), i \in \{1, 2, 3, \dots, 30\}$

Fig. 2. Schematic diagram of K-fold Cross-Validation

As the above figure shows, the SSE is lowest when $k = 5$, hinting us that 5 neighbors will give us the least error between estimated positions and actual positions.

The above 3 figures show the floor plan view of our dataset, the grids are the actual rooms, the gray points are the training dataset, the black points are the testing dataset, the black squares being the access points, and the black lines indicates the error between the actual and estimated positions.

We can see that $k = 3$ shows a much smaller error in distance than $k = 1$. While the plot for $k = 5$ is similar to the plot for $k = 1$, the average and median error in distance is actually smaller in $k = 5$.

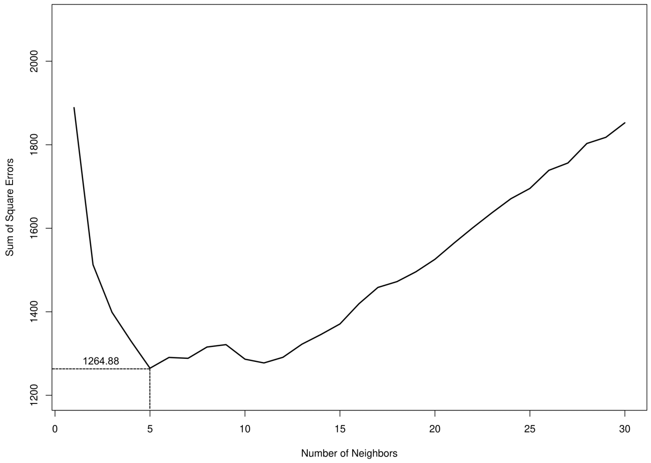


Fig. 3. Schematic diagram of K-fold Cross-Validation

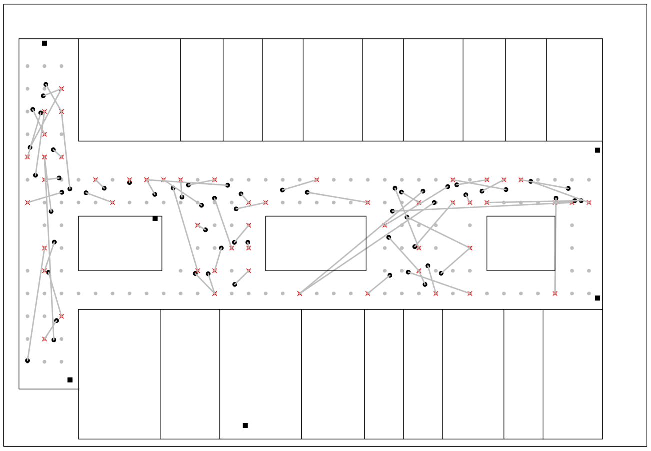


Fig. 4. Floor plan of estimated device locations when $k = 1$

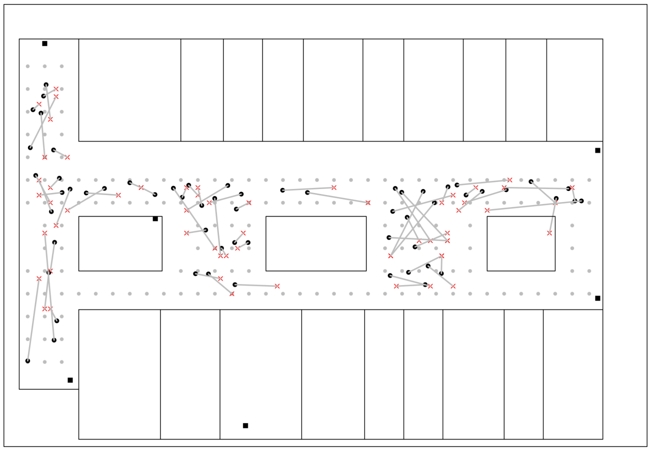


Fig. 5. Floor plan of estimated device locations when $k = 3$

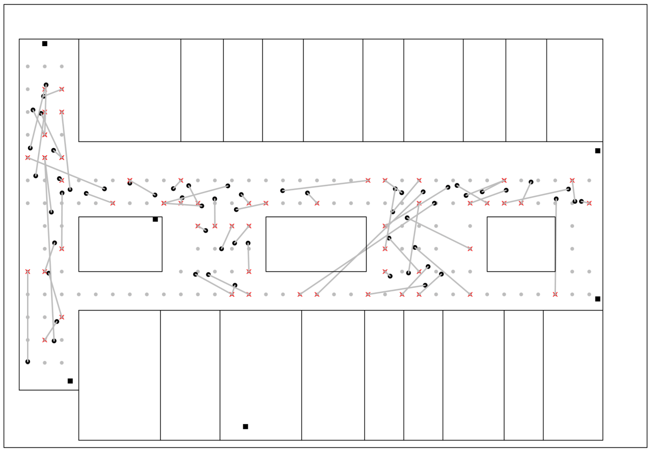


Fig. 6. Floor plan of estimated device locations when $k = 5$

of k	Average Error Distance	Median Error Distance
$k = 1$	2.517842 m	1.902775 m
$k = 3$	1.931918 m	1.662997 m
$k = 5$	1.769582 m	1.412887 m

IV. Evaluation

V. Conclusion

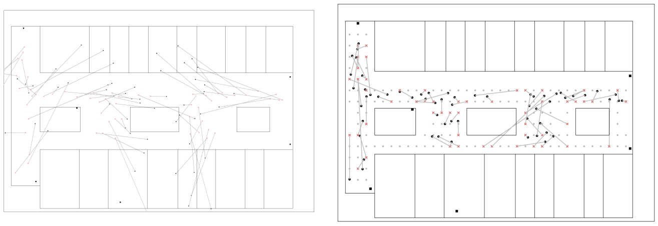
- Extended version of the executive

VI. Summary

- Contextualization of results in light of project goals
- Make explicit connection to project background by following the implications of our results

References

[1]



(a) Trilateration Prediction

(b) KNN Prediction

Fig. 7. Error of distance using 2 methods

Method	Average Error Distance	Median Error Distance
Trilateration	4.832989 m	4.955922 m
KNN	2.517842 m	1.902775 m

[2]
[3]
[4]
[5]
[6]
[7]
[8]
[9]
[10]

Appendix A: Data Tables

Appendix B: Derivation of Equations

Appendix C: Index