Enhanced Indoor Locationing in a Congested Wi-Fi Environment

Hsiuping Lin, Ying Zhang, Martin Griss, Ilya Landa
Carnegie Mellon Silicon Valley
{tony.lin, joy.zhang, martin.griss, ilya.landa} @sv.cmu.edu

MRC-TR-2009-04
March 2009

ABSTRACT

Many context-aware mobile applications require a reasonably accurate and stable estimate of a user's location. While the Global Positioning System (GPS) works quite well world-wide outside of buildings and urban canyons, locating an indoor user in a real-world environment is much more problematic. Several different approaches and technologies have been explored, some involving specialized sensors and appliances, and others using increasingly ubiquitous Wi-Fi and Bluetooth radios. In this project, we want to leverage existing Wi-Fi access points (AP) and seek efficient approaches to gain usefully high room-level accuracy of the indoor location prediction of a mobile user. The Redpin algorithm, in particular, matches the Wi-Fi signal received with the signals in the training data and uses the position of the closest training data as the user's current location. However, in a congested Wi-Fi environment where many APs exist, the standard Redpin algorithm can become confused because of the unstable radio signals received from too many APs. In this paper, we propose several enhanced indoor-locationing algorithms for the congested Wi-Fi environment. Different statistical learning algorithms are compared and empirical results show that: using more neighbors gives better results than using the 1-best neighbor; weighting APs with the correlation between the AP visibility and the location is better than the equally weighted AP combination, and automatic filtering noisy APs increases the overall detection accuracy. Our experiments in a university building show that our enhanced indoor locationing algorithms significantly outperform the-state-of-the-art Redpin algorithm. In addition, this paper also reports our findings on how the size of the training data, the physical size of the room and the number of APs affect the accuracy of indoor locationing.

Categories and Subject Descriptors

General Terms

Algorithms, Measurement, Experimentation

Keywords

Wi-Fi, context-aware computing, location-based services, Wi-Fi signatures, mobile computing, proximity measurement

1. INTRODUCTION

Location information is a crucial input to many context-aware mobile applications both indoors and out, such as personal navigation, asset tracking, local information searches and friend finder. While the Global Positioning System (GPS) works quite well outside, it does not work well inside most buildings because of weak or inaccessible satellite signals. We desire a stable and accurate indoor location system for use at home, in the office, and other buildings. Therefore, Indoor Locationing plays an important role in ubiquitous computing and attracts considerable interest in both industry and research. Some systems are designed specifically for indoor locationing but require special infrastructure [1] [2]. With the growth of Wi-Fi networks due to declining prices, increased ubiquity of devices (laptops, cell phones, and other devices using Wi-Fi) and simplified installation of Wi-Fi access points (AP), indoor locationing using Wireless LAN (WLAN) is becoming more promising.

Two common categories of techniques are used in WLAN indoor locationing. One uses a signal propagation model and geometric information on the placement of APs to convert the received signal strength (RSS) into a distance measurement and to estimate the physical location using a multi-lateration algorithm. Some of these methods have become quite sophisticated, with models for wall-attenuation, and reflections. They also scale well in terms of installation and calibration.[1] However, this approach does not always provide satisfying results because environmental variations cause significant fluctuations in Wi-Fi signals in the same location over time [3]. The other approach uses location fingerprints which consist of a set of RSSes collected from the APs associated with a set of location symbols in the environment. The first step of this technique is to establish a fingerprints database. Then the location of one mobile device can be decided by comparing its current fingerprint to the historical fingerprints in the database to see which pair has the highest similarity [4].

Many approaches have been tried with location fingerprinting. The simplest is the K-nearest neighbor algorithm (KNN). It converts fingerprints into vectors and chooses the K historical fingerprints with the shortest distance to the measured fingerprint to estimate the location. Another form of KNN uses a similarity value as the distance. It measures not only the contribution of RSSes but also the number of common access points and not-common access points [8]. Another very popular approach is the probabilistic method. It is based on an empirical model which describes the distribution of RSS at various location symbols and tries to handle the uncertainty and errors of signal strength measurements [5]. In addition, Statistical Learning Theory is also adopted by this technique to derive the unknown functional dependency on the basic observation [9].

In this paper, we combine the benefits of KNN and statistical history of fingerprints to provide a simple but effective indoor locationing approach to

[1] See products from Cisco, and from Motorola, and discussion by Correa et al [13].

achieve high room-level accuracy, especially in a congested Wi-Fi environment. The rest of this paper is organized as follows. Section 2 briefly reports on previous work concerning indoor location estimation. Section 3 lists common approaches to indoor locationing and proposes new approaches. Section 4 describes the dataset, our experimental environment and key results. Section 5 draws conclusions and suggests future work.

2. RELATED WORK

A reliable and stable Interior Locationing system would be of great benefit to many applications, and thus considerable research has been performed to determine the indoor location of a mobile user or a mobile device. RADAR, developed by Bahl, is a famous indoor location estimation system based on Wi-Fi technology [1]. It collects device RSS fingerprints at the APs and calculates the mobile user's location by empirical methods and by signal propagation modeling. Most recent research instead collects RSSes directly at the mobile devices, avoiding the need for extra hardware elements. Li [5] discusses the comparison of the trilateration and fingerprinting approaches, including both deterministic methods and probabilistic methods. Brunato [9] provides a general comparison of SVM, KNN, Bayesian modeling and multi-layer perceptrons. Carlotto [7] evaluates the proximity of two mobile devices by classifying the degree of similarity of the Wi-Fi scanned data using a statistical Gaussian Mixture Model. Correa [13] reports experiences using an existing Wi-Fi infrastructure without specialized hardware added to support room-level Wi-Fi location tracking by signature matching, as well as the use of a specialized AP controller. Bolliger [8] proposes a novel approach in his Redpin system to fingerprinting that does not require an explicit offline phase but lets users in the system create and manage the location fingerprints in a collaborative way. In our work, we initially started from the open source Redpin system, but have made substantial enhancements based on our research.

3. APPROACH

Our location fingerprinting is based on the heuristic that a mobile user will experience a different RSS fingerprint at different locations in the building, and that the variation of the fingerprints seen over time in one location does not vary too much[2]. In this section, we discuss the algorithms we used in the following experiments. We also suggest a noise filter to reduce the impact from signal fluctuations and improve the accuracy.

3.1 Naïve Bayes Classifier

The naïve Bayes classifier is based on Bayes theorem with a naïve independence assumption. If we want to know the most likely location L_x by a

[2] Of course, the usefulness of this location-based difference and relatively stable fingerprint depends on the placement of the access points, the shape and construction of the building and the sources of noise and fluctuation.

given fingerprint **AP**=*(AP₁, … APₙ),* we need to know the joint probability of L_x and **AP**, which is

$$\begin{aligned} P(L_x, AP_1,..., AP_n) &= P(L_x)P(AP_1,..., AP_n \mid L_x) \\ &= P(L_x)P(AP_1 \mid L_x)P(AP_2 \mid L_x, AP_1)\bullet\bullet\bullet P(AP_n \mid L_x, AP_1,..., AP_{n-1}) \\ &= P(L_x)P(AP_1 \mid L_x)P(AP_2 \mid L_x)\bullet\bullet\bullet P(AP_n \mid L_x) \\ &= P(L_x)\prod_{i=1}^{n} P(AP_i \mid L_x) \end{aligned}$$

Then the conditional probability of L_x given AP is

$$P(L_x \mid AP) = \frac{1}{Z} P(L_x)\prod_{i=1}^{n} P(AP_i \mid L_x)$$

Z is a constant which only depends on AP_1, …, AP_n. We can easily get the probability of the measured fingerprint for each location and pick the highest probability as our prediction.

The problem of the naïve Bayes classifier applying to indoor positioning is that the values of the signal strength are not taken into consideration. Only the existence of a set of APs decides the location. To address this, technologies such as particle filters can be introduced to improve the accuracy [6].

3.2 Support Vector Machine

The Support Vector Machine (SVM) is a useful technique for data classification and some research has applied SVM to the indoor locationing problem [9] [10]. A classification task usually involves training and testing data which consist of many data instances. Each instance in the training set contains one target value (class labels) and several attributes (features). The goal of SVM is to produce a model which predicts the target value of data instances in the testing set when given only the attributes. LIBSVM (A Library for Support Vector Machines) is chosen as the tool to infer the locations of the measured fingerprints [11] [12].

Although SVM is a powerful classification technique, it needs offline training. Another problem is the fluctuations of signals may cause data instance pollution and reduce the accuracy.

3.3 K-Nearest Neighbor

The K-nearest neighbor algorithm is a method for classifying objects based on the closest distances and a majority vote of the nearest neighbors. In the simplest case (K=1), the algorithm finds the single closest match.

3.3.1 Vector Distance

The standard K-nearest neighbors algorithm goes through each historical fingerprint vector $\mathbf{Y_i} = [Y_{i1}, Y_{i2}, \ldots, Y_{iN}]$ in the database and calculates its distance to the measured fingerprint vector $\mathbf{X} = [X_1, X_2, \ldots, X_N]$ where X_j and Y_{ij} are observed RSS belonging to the unknown location **X** and known location $\mathbf{Y_i}$, and N is the total number of APs. The generalized distance is

$$D_q(X, Yi) = \left(\sum_{j=1}^{N} | X_j - Y_{ij} |^q \right)^{\frac{1}{q}}$$

Manhattan distance and Euclidean distance are D_1 and D_2 respectively. The unknown location **X** is decided by a majority vote from the K shortest distance fingerprints. Similar to KNN, WKNN using an inverse distance weighted average is also introduced to improve the accuracy.

KNN is simple to implement and provides reasonable accuracy; however, one drawback of standard KNN in indoor locationing is that the RSSes detected in the same location vary from time to time, which may cause measurement error.[3] The other is that some information available in the historical fingerprints may be ignored since this algorithm solely depends on the value of signal strength.

3.3.2 Redpin Algorithm - AP Similarity

In addition to the effect of signal strength, the number of common access points (NCAP) and the number of not-common access points (NNAP) also contribute to identifying the similarity of two fingerprints. Unlike finding the closest distance in KNN, we want to find the highest similarity of the fingerprints. The Redpin algorithm uses a weighted combination of the Vector distance and the AP similarity and chooses K=1 to decide the best match. The Redpin source code can be found here [14]. The Redpin algorithm works as follows.

First of all, a mapping function is defined as

$$\delta(X_i) = \begin{cases} 0, & X_i = 0 \\ 1, & X_i \neq 0 \end{cases}$$

NCAP of two fingerprints X, Y can be expressed as

$$NCAP = \sum_{i=1}^{N} (\delta(X_i) \bullet \delta(Y_i)) \text{ , where } \bullet \text{ represents the dot product}$$

NNAP of X, Y can be expressed as

$$NNAP = \sum_{i=1}^{N} (\delta(X_i) \oplus \delta(Y_i)) \text{ , where } \oplus \text{ represents the exclusive disjunction}$$

The generalized similarity value of X, Y is

[3] This is partially overcome by having multiple fingerprint sets for a given location, taken at different times; the assumption is that one or other finger print may cover that fluctuation.

$$S(X,Y) = \alpha \times \sum_{i=1}^{N} (\delta(X_i) \bullet \delta(Y_i)) - \beta \times \sum_{i=1}^{N} \delta(X_i) \oplus \delta(Y_i) + \gamma \times C(X,Y)$$

C is a function which calculates the contribution of signal strengths from X and Y. α and γ are the bonus-weight for the common APs while β is the penalty-weight for the not-common APs. By introducing NCAP and NNAP and bonus-penalty adjustments, the impact of signal fluctuations can be reduced.

3.3.3 Correlation between Locations and APs

Since environmental variations cause significant Wi-Fi signal fluctuations in the same location over time, the visibility of the APs to the same location is not always the same, especially inside a large building with sparse APs. For example, AP_1 and AP_2 may be common access points to location Z but they could different visibility distributions to Z. It may not make sense for AP_1 and AP_2 to have the same bonus when calculating the similarity value. To evaluate the correlation between one access point and one location, we introduce Point-wise Mutual Information (PMI), defined as

$$I(Loc_k; AP_i) = \log \frac{P(Loc_k, AP_i)}{P(Loc_k)P(AP_i)}$$

The greater $I(Loc_k; AP_i)$ is, the more likely Loc_k is associated with AP_i. From the historical fingerprints in the database, we can easily calculate the $I(Loc_k; AP_i)$ value of each location for each AP and normalize the value to be between 0 (the lowest correlated) to 1 (the highest correlated). Then PMI values can be applied as weighted modifiers to the bonus of each common AP (CAP) and the penalty of each not-common AP (NAP). The similarity value with PMI is

$$S(X,Y) = \alpha \times \sum_{j=1}^{n} I(X;Y_{cj}) - \beta \times \sum_{j=1}^{m} I(X;Y_{ncj}) + \gamma \times \sum_{j=1}^{n} C(X,Y_{cj})$$
$$Y_c \in \{CAP\}, Y_{nc} \in \{NAP\}$$

3.3.4 Noise Filter

Inspired by PMI, since not all APs have the same contribution to one location, we define a noise filter to remove less frequently visible APs in the fingerprints of one location, essentially regarding these signals as noise. The remaining APs of the fingerprints are very relevant to the location and we call them “relevant AP”. The average frequency of all visible APs in each fingerprint (X_j) of the location (Loc_k) in the training data is calculated by

$$\overline{F}(Loc_k) = \frac{1}{N} \sum_{i=1}^{N} F(Loc_k, i)$$, where

$$Z(X,i) = \begin{cases} 0, & X_i = 0 \\ 1, & X_i \neq 0 \end{cases}$$, and

$$F(Loc_k, i) = \sum_{j=1}^{M} Z(X_j, i), \quad X_j \in \{Loc_k\}$$

The noise filter can be expressed as

$$N(X) = X', \quad where \quad X'_i = \begin{cases} 0, & F(Loc_k, i) < \overline{F}(Loc_k) \\ X_i, & F(Loc_k, i) >= \overline{F}(Loc_k) \end{cases} \quad and \quad X \in \{Loc_k\}$$

4. EXPERIMENT

4.1 Environment

The target system for the experiments is a wireless LAN using the IEEE 802.11b (Wi-Fi) standard. The LAN is composed of seven 3-COM APs and six Motorola APs. These APs are located on both the first and second floors of our building. We have also adjusted the locations of these APs to give each room and area reasonable average coverage. In addition, there are another three APs outside our building that could also be detected in our experiments. These 16 APs make our building become a congested Wi-Fi environment. Moreover, we used Nokia N95 smart phones with built-in Wi-Fi 802.11 b/g to collect fingerprints. The target environment is on the second floor, roughly 60mx15m with a 15mx12m lounge. The maps of the both floors and the locations of APs are shown in Figures 1 and 2. Red dots in these figures are the locations of APs.

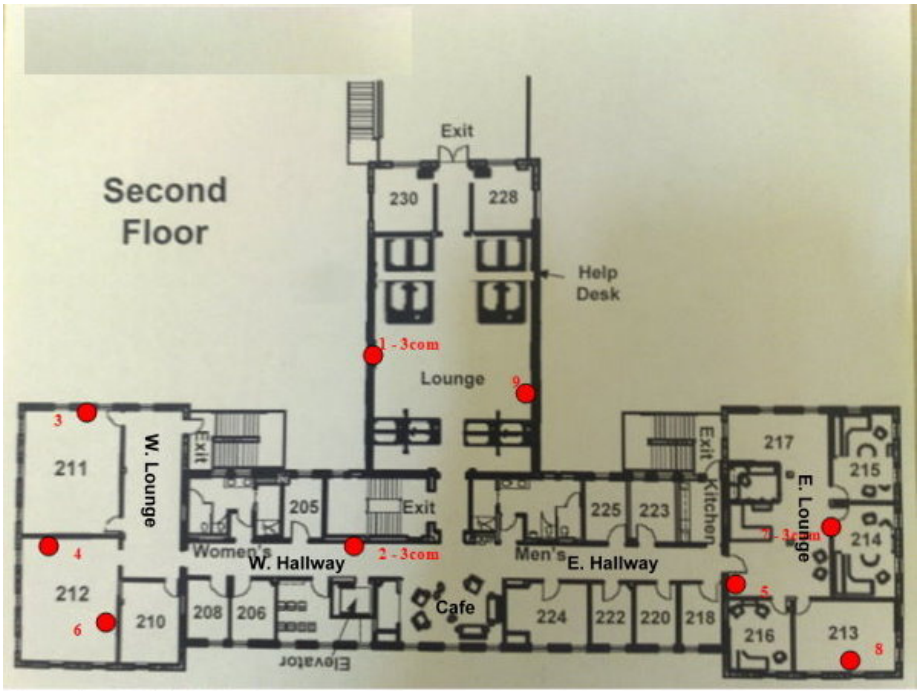

Figure 1: The map of the second floor

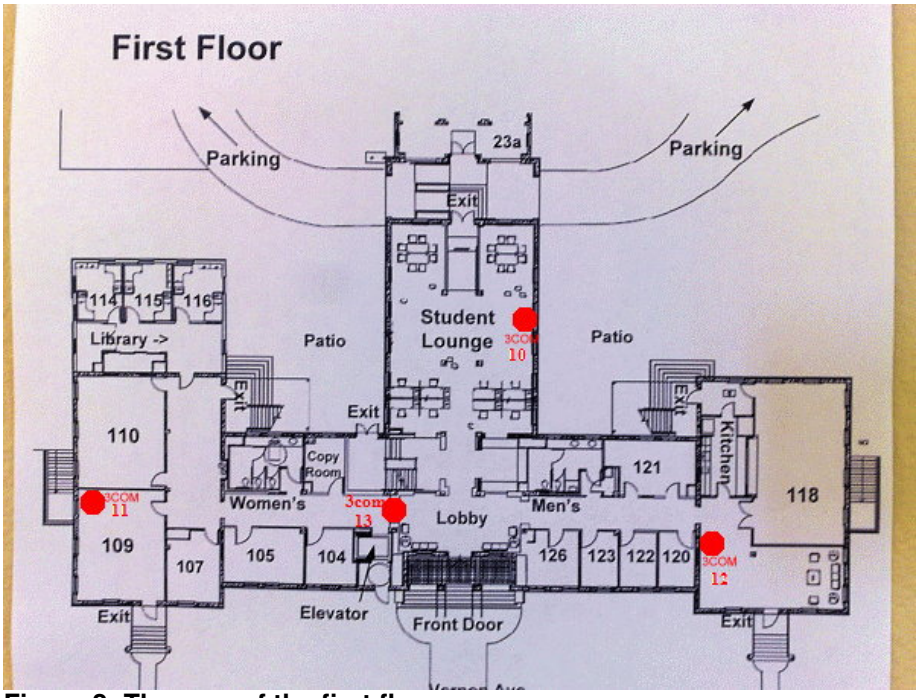

Figure 2: The map of the first floor

In the first-stage experiments, we selected 9 public rooms on the second floor, and collected over 100 fingerprints from each room over a period of several days. The total number of unique fingerprints is 1,002. A text file containing all the fingerprints can be found at http://hplin.tw.googlepages.com/data.csv. The fingerprint distribution is shown on Table 1.

Rm211	Rm212	W lounge	W hallway	Café	Lounge	E hallway	E lounge	Rm213
150	100	125	101	101	100	100	124	101

Table 1: The fingerprint distribution for each room

4.2 Experimental Methods

To effectively analyze the various experiments, we use stratified 10-fold cross-validation to evaluate the accuracy. Also we ran 100 trials, each randomly choosing 10% of all historical fingerprints as testing data with the remaining 90% as training data to measure the confidence interval. The algorithms we used are listed as follows:

- **Naïve Bayes Classifier**
  We build nine naïve Bayes classifiers, one for each room and then calculated nine possibilities for one test fingerprint, picking the highest one as our prediction.
- **Support Vector Machine**
  We scale both training data and testing data, and run iterative cross validations to find the best parameters for the cost and gamma values for LIBSVM prediction.

- **K-Nearest Neighbor**
  We calculate both the Manhattan distance and the Euclidean distance from K=1 to K=9. We found that Manhattan distance and K=5 generates the best results (see section 3.3.1).
- **Redpin**
  We use the default Redpin algorithm with $\alpha = 1$, $\beta = 0.4$, $\gamma = 0.2$ and the value of C function limited from -1 to 1, depending on the contribution of the signal strength of two common APs (see section 3.3.2).
- **Enhanced Redpin**
  The enhanced Redpin is similar to Redpin except we choose K=5 to gain better accuracy and add point-wise mutual information (PMI) as the modifier for each access point to each location (see section 3.3.3). PMI values could easily be calculated from historical data.

4.3 Result

The result of each algorithm is shown in Table 2 and the accuracy of KNN, Redpin and the enhanced Redpin for K from 1 to 10 is shown in Table 3. It is not surprising that naïve Bayes classifier gets the worst accuracy since the rooms are quite close to each other so that the existence of a particular set of APs was not as dominant a factor as the difference in signal strengths. Moreover, SVM and KNN have similar accuracy because they both depend highly on the signal strength information. As we expect, the Redpin algorithm has better performance than KNN because it reduces the signal fluctuations by introducing NCAP and NNAP and bonus-penalty adjustments. Also, the enhanced Redpin algorithm gets the best result, outperforming the original Redpin by 6%. The 95% accuracy confidence interval difference between the enhanced Redpin and Redpin is from 0% to 15%. Moreover, from Table 3, the improvement by PMI modifiers is consistently around 1%. Though not immediately apparent, the correlation between locations and APs does contribute to the accuracy. More research on alternative statistical methods for the correlation is planned for the future.

	Naïve Bayes	SVM	KNN	Redpin	Enhanced Redpin
Accuracy	61%	80%	79%	81%	87%
Confidence interval (95%)	54% - 68%	75% - 86%	71% - 85%	76% - 88%	82% - 92%

Table 2: Accuracy of each algorithm

	K=1	K=2	K=3	K=4	K=5	K=6	K=7	K=8	K=9	K=10
KNN	78%	78%	79%	80%	79%	79%	78%	79%	78%	78%
Redpin	81%	81%	83%	85%	86%	86%	85%	86%	86%	85%
Enhanced Redpin	82%	82%	84%	85%	87%	87%	87%	87%	86%	86%

Table 3: Accuracy of Redpin and Enhanced Redpin (K from 1 to 10)

The room-level accuracy is shown in Figure 3 and the distribution of APs is listed in Table 4. We observe two interesting results. One is that the naïve Bayes classifier gains the highest accuracy in Room213. The most plausible

explanation is that only six APs are visible in Room213 and even the room with the least visible AP has 35% accuracy. The other result is that all algorithms had highest accuracy in the Lounge. The reasonable explanation is that Room Lounge is the largest room so the estimate error is less significant.

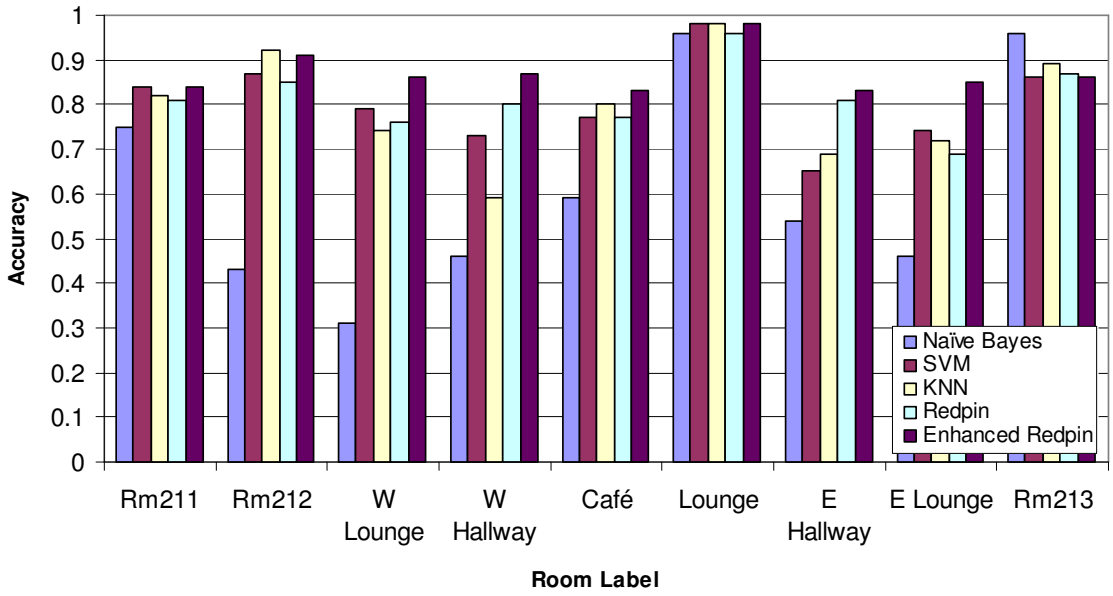

Figure 3: Room-level accuracy of each algorithm

	Rm211	Rm212	W Lounge	W Hallway	Café	Lounge	E Hallway	E Lounge	Rm213
AP1	0%	1%	5%	26%	52%	61%	52%	3%	0%
AP2	16%	1%	42%	69%	99%	100%	79%	18%	0%
AP3	0%	0%	1%	34%	95%	100%	50%	2%	0%
AP4	65%	98%	99%	100%	99%	95%	100%	84%	61%
AP5	8%	0%	1%	0%	0%	18%	4%	12%	0%
AP6	43%	62%	82%	71%	83%	44%	66%	15%	0%
AP7	57%	94%	89%	77%	78%	8%	62%	44%	38%
AP8	1%	23%	13%	46%	39%	1%	49%	85%	96%
AP9	5%	39%	34%	60%	58%	6%	62%	94%	81%
AP10	66%	98%	100%	87%	36%	10%	34%	2%	0%
AP11	65%	97%	99%	85%	37%	0%	40%	15%	0%
AP12	1%	6%	2%	18%	23%	0%	17%	28%	34%
AP13	0%	1%	1%	28%	26%	0%	78%	97%	98%
AP14	65%	95%	97%	78%	10%	0%	19%	1%	0%
AP15	0%	0%	0%	0%	1%	2%	0%	0%	0%
AP16	0%	0%	0%	0%	8%	3%	0%	0%	0%

Table 4: The distribution of APs for each room (The relevant AP of each room is marked as a light blue background)

4.4 Impact of Training Data Size

To evaluate the precision of the five algorithms for different size of historical fingerprints, we varied the size from 50 to 900 in increments of 50 and ran

stratified 10-fold cross-validation. The accuracy of different fingerprint size for each algorithm is shown in Figure 4. Even with a small training data size (150), the enhanced Redpin algorithm can provide over 80% accuracy, which is better than other alternatives. However, when the number of historical fingerprints is increased, the accuracy improvement is less apparent. One plausible reason is that while more fingerprints provide more matching samples, they also provide more polluted data, confusing the algorithms and reducing the accuracy.

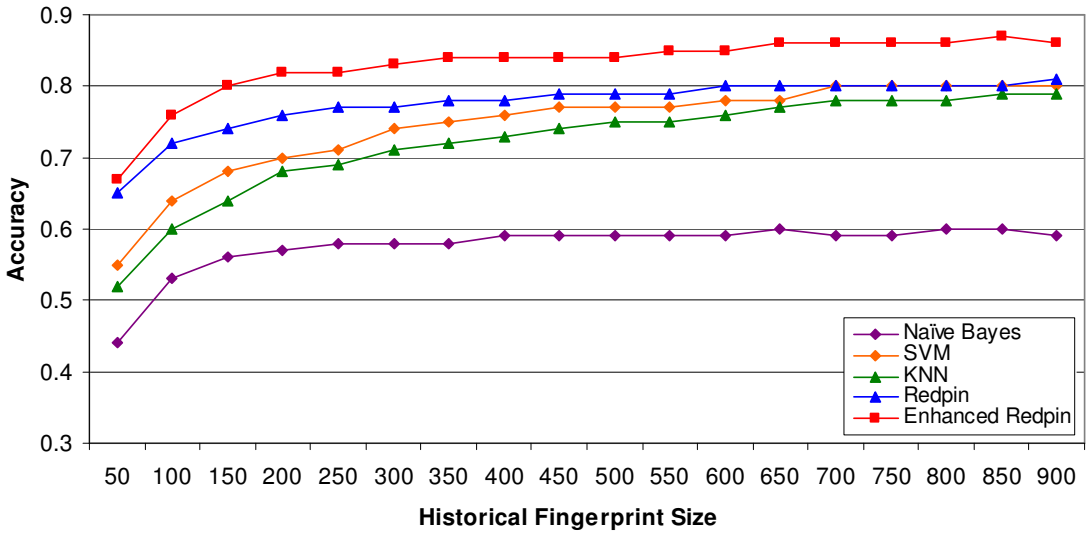

Figure 4: Accuracy of different historical fingerprint size

4.5 Granularity of Rooms

To see the effect of room granularity on accuracy, we combine one room with its adjacent rooms to create a larger virtual "room". For example, if the testing fingerprint does not match Room211, the neighboring rooms such as Room212, W Lounge and W Hallway would be checked to see if any room is matched. The adjacent rooms we use are shown in Table 5 and the order of the adjacent room is from 1 to 3. The accuracy of each virtual adjacent area is shown in Figure 5. When one adjacent room is added, the accuracy of all algorithms can be enhanced to over 80% and when four adjacent rooms are combined, the accuracy can even be better than 90%. Therefore all algorithms gain high accuracy when estimating a coarse-grained location. However, for finer-grained locations, the enhanced Redpin algorithm is the most accurate.

0	Rm211	Rm212	W Lounge	W Hallway	Café	Lounge	E Hallway	E Lounge	Rm213
1	Rm212	Rm211	Rm211	W Lounge	Lounge	Café	E Lounge	E Hallway	E Lounge
2	W Lounge	W Lounge	Rm212	Café	W Hallway	W Hallway	Café	Rm213	E Hallway
3	W Hallway	W Hallway	W Hallway	Lounge	E Hallway	E Hallway	Lounge	Café	Café

Table 5: Adjacent rooms for each room

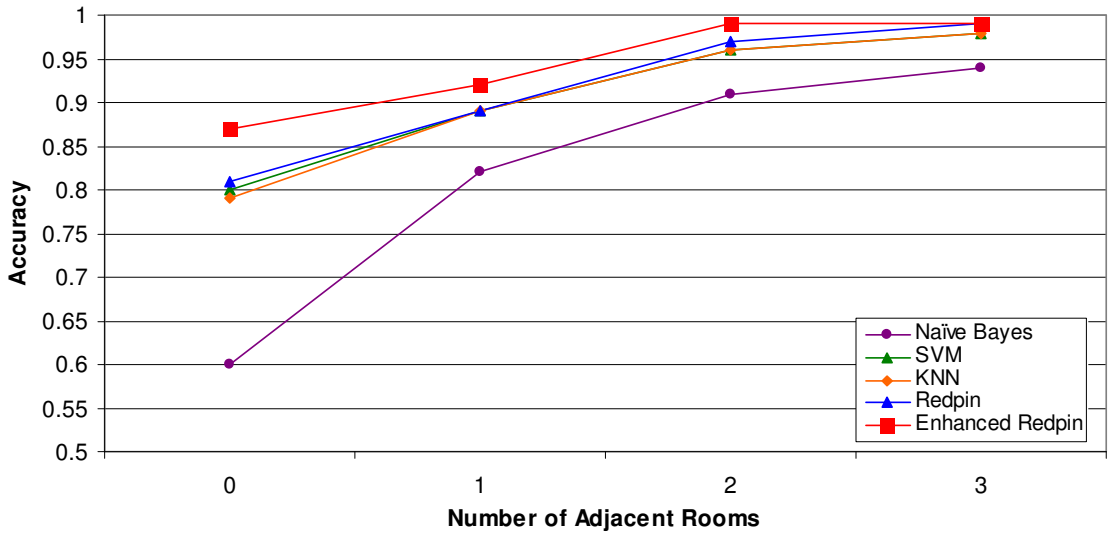

Figure 5: Accuracy of virtual rooms

4.6 Noise Filter

In the previous experiments, all detected RSSes of the APs are measured and used in the calculation. To see the impact of the signal noise in the congested Wi-Fi environment, we apply the noise filter on the training data set (see section 3.3.4 and Table 4: The distribution of APs). Table 6 lists the maximum number of the original APs and the relevant APs in the fingerprints of each location.

	Rm211	Rm212	W Lounge	W Hallway	Café	Lounge	E Hallway	E Lounge	Rm213
Original APs	11	12	14	13	15	12	14	14	6
Relevant APs	6	7	7	8	5	5	9	5	6

Table 6: Maximum number of the original APs and the relevant APs

4.6.1 Result

First of all, we repeat the same procedure as the first experiment except we keep the testing data as raw data and apply the noise filter on the training data. The accuracy for each algorithm is listed in Table 7. We add Redpin (K=5) in this table to see the effect of PMI modifiers. From the result, all of the algorithms except naïve Bayes classifier improve their accuracy because most of signal noise is filtered out. In addition, PMI did not provide benefits for Redpin (K=5) because the correlation between the locations and the relevant APs is already strong enough, which makes PMI modifiers almost useless.

	naïve Bayes	SVM	KNN	Redpin	Redpin (K=5)	Enhanced Redpin
Accuracy without filter	61%	80%	79%	81%	86%	87%
Accuracy with filter	64%	86%	88%	88%	90%	90%
Confidence interval (95%)	59% - 73%	80% - 92%	84% - 94%	84% - 94%	84% - 95%	86% - 96%

Table 7: Accuracy of each algorithm after applying the noise filter (see first and second rows)

4.6.2 Impact of Different Filtered Training Data Size

From the previous experiment, the noise filter can enhance the accuracy of all algorithms. However, does the noise filter still work when the training data size is small? To evaluate the impact of the different filtered training data size, we repeat the same experiment as section 4.4 except we keep the testing data as raw data and apply the noise filter on the different size of training data. The result is shown in Figure 6. Compared to Figure 4, the accuracy of different training data size, we only need 150 filtered fingerprints to make all algorithms except naïve Bayes classifier to reach about 80% accuracy. Four algorithms can gain over 80% accuracy with 250 filtered fingerprints. After about 300 fingerprints, no further improvement is seen. The noise filter can successfully reduce the signal fluctuations and benefit most algorithms.

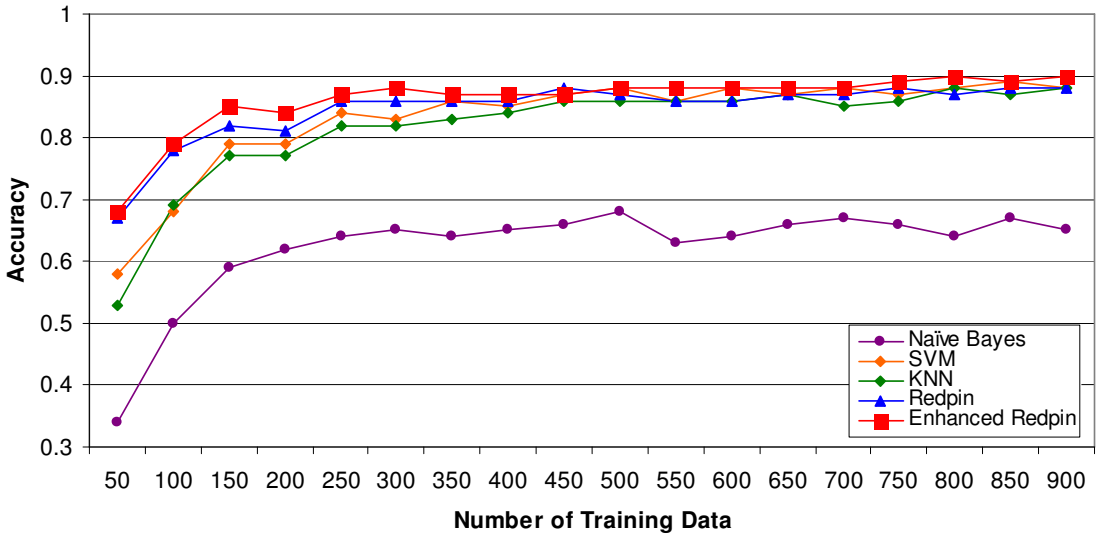

Figure 6: Accuracy of different filtered training data size

4.6.3 Number of Relevant APs

Next, we want to see how many relevant APs for one room are needed for acceptable accuracy. The APs seen by each room are sorted from the most relevant to the least. We run this experiment eight times, increasing the number of relevant APs in each experiment, while ensuring that we never exceed the

number of maximum APs of each room. For the training data, it is easy to apply the noise filter since the locations were recognized. However, we need to assume that the testing fingerprint is located at the same location as the training fingerprint. When running the experiment, we filter out irrelevant APs in the testing fingerprint based on the location of each training fingerprint. Because of this, it is difficult to measure the accuracy of SVM since the probability values of different training models calculated by LibSVM were not correlated. The accuracy of different numbers of APs for the other four algorithms is shown in Figure 7. From the result, in small numbers of APs (from two to four), the Redpin algorithm has much better accuracy than KNN and enhanced Redpin. One plausible explanation is that the bonus-penalty adjustment of Redpin makes the discrepancy of two fingerprints more obvious but the PMI modifiers cannot provide any additional benefit. Sometimes PMI even reduces the effect of bonus-penalty adjustment because the filtered APs are highly relevant to the locations.

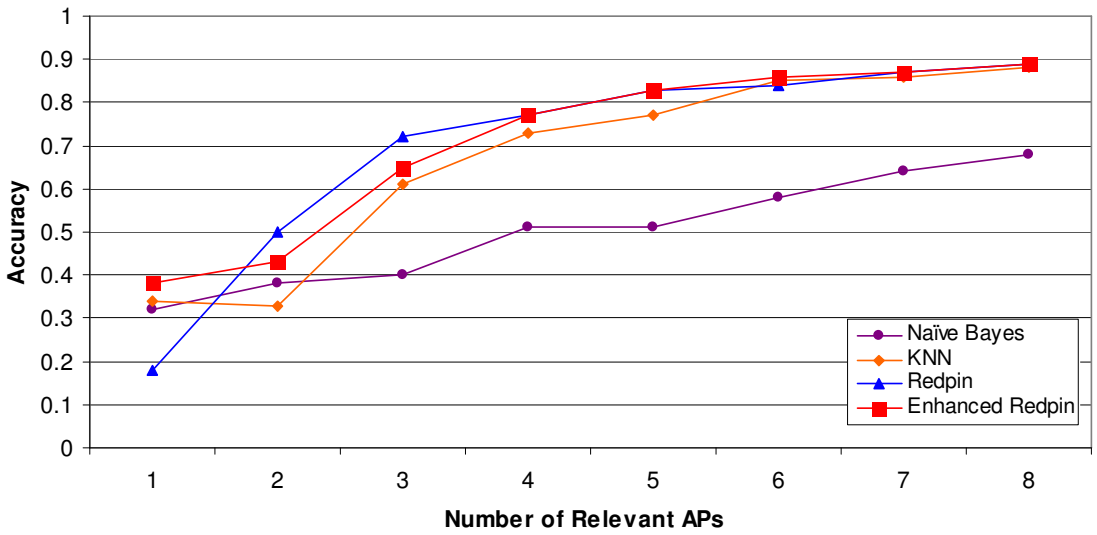

Figure 7: Accuracy of different number of relevant APs

4.7 Number of APs

To extend the experiment of different number of relevant APs for each room, we evaluated how many APs can contribute acceptable accuracy for our building. First of all, we sort the APs by their visibility from the highest frequency to the lowest one. For each run, we increase the number of APs from 1 to 16 to calculate the accuracy. The accuracy for each algorithm is shown in Figure 8. We see that SVM has better accuracy when the number of APs is fewer than ten because the primary APs can provide enough information for the SVM classification. However, when the number is more than ten, the SVM classification is affected by fingerprint pollution and the result is worse. On the other hand, the Redpin algorithm performs better than enhanced Redpin while the number is fewer than three. But when the number is more than ten, PMI does help Redpin to handle the signal noise. Another interesting observation is that the improvement of accuracy slows after nine APs. It seems that these less visible

APs do not provide essential information for location detection. Instead, they may even cause confusion when matching fingerprints and reduce the accuracy.

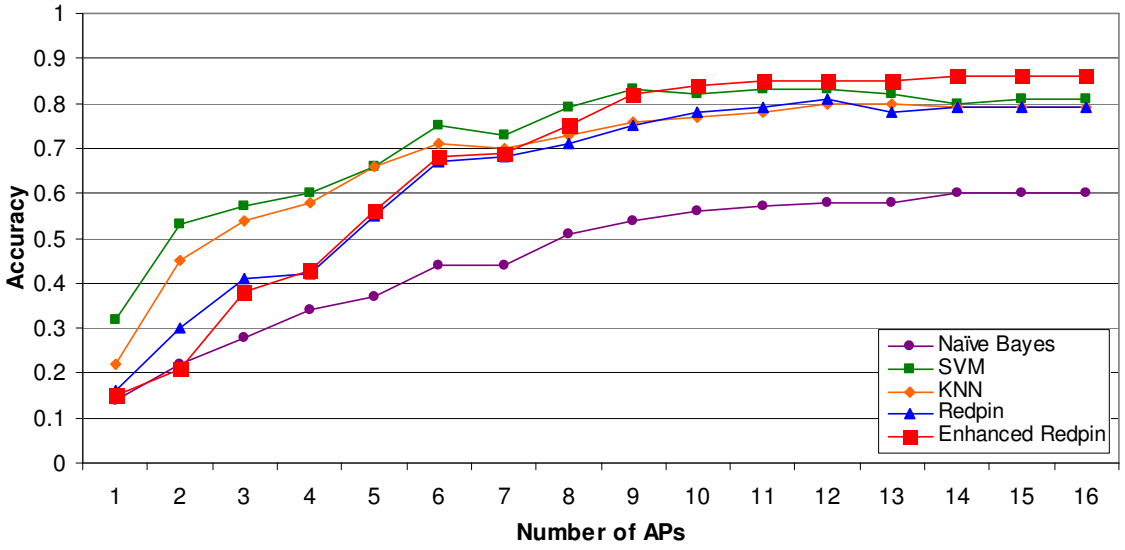

Figure 8: Accuracy of different number of APs

5. CONCLUSION AND FUTURE WORK

In this paper, the common techniques used in indoor locationing using fingerprints are discussed and evaluated. Our proposed approach takes signal strengths, AP visibility and statistical fingerprint history into consideration to enhance the Redpin algorithm in a congested Wi-Fi environment. This approach obtains the best accuracy and also works well even with the small training data set in the experiments. By applying the noise filter to detect user's location, the enhanced Redpin algorithm may not work well with a small number of APs. However, most office buildings and homes in our area are covered by more than three APs. The fluctuations and congested signals are likely to be more serious at home than in the laboratory or office. We believe the enhanced Redpin algorithm with the noise filter should be able to provide an overall satisfying indoor locationing prediction.

In the first-stage experiment environment, we only choose nine public rooms on the second floors. We plan to extend the collection to more private and wall-bounded rooms over two floors. Multiple RF fingerprints, such as (muted) Bluetooth, might also improve the accuracy[4]. Finally, we will explore use of accelerometer data to determine if a user is moving or not and thus enable time-averaging or tracking to improve accuracy. Another interesting issue is to evaluate how many APs and how to locate them to produce good enough accuracy for one building. Finally, we plan to apply and evaluate our improved Interior room-level location to several mobile health and mobile professional scenarios.

[4] The original Redpin used Bluetooth and GSM cell-ID to augment Wi-Fi

6. ACKNOWLEDGEMENTS

We are grateful for support provided by CyLab and Nokia, and for the initial assistance of Philip Bolliger to get us started using Redpin. We also acknowledge the efforts of Joshua Correa and Ed Katz, who explored an earlier set of approaches to Wi-Fi based location in our building [13], to Patricia Collins who reviewed multiple versions of the paper.

7. REFERENCES

[1] Bahl, P. and Padmanabhan, V.N., "RADAR: an in-building RF-based user location and tracking system", IEEE INFOCOM 2000, Mar. 2000

[2] Hightower, J. and Borriello, G., "Location systems for ubiquitous computing", IEEE Computer, Aug. 2001

[3] Ho, W., Smailagic, A., Siewiorek, D.P. and Faloutsos, C., "An adaptive two-phase approach to WiFi location sensing", IEEE Int. Conf., Mar. 2006

[4] Barceló, F., Evennou, F., de Nardis, L. and Tomé, P., "Advances in indoor Location", TOPO-CONF-2006-024, 2006

[5] Li, B., Salter, J., Dempster, A.G., and Rizos, C.,"Indoor positioning techniques based on Wireless LAN" IEEE Int. Conf., Mar. 2006

[6] David Madigan, D., Elnahrawy, E.,. Martin, R.P, Wen-Hua J., Krishnan, P. and Krishnakumar, A. S. "Bayesian Indoor Positioning Systems", IEEE Infocom, Mar. 2005

[7] Carlotto, A., Parodi, M., Bonamico, C., Lavagetto, F., and Valla, M., "Proximity classification for mobile devices using wi-fi environment similarity", ACM International Workshop, 2008

[8] Bolliger, P. "Redpin - adaptive, zero-configuration indoor localization through user collaboration", ACM International Workshop, 2008

[9] Brunato, M., and Battiti, R., "Statistical Learning Theory for Location Fingerprinting in Wireless LANs", Computer Networks, 2005

[10] Fan, R.E. , Chen, P.H. and Lin, C.J. "Working set selection using the second order information for training SVM", Journal of Machine Learning Research 6, 1889-1918, 2005

[11] Chang, C.C, and Lin, C.L, LIBSVM : a library for support vector machines, 2001. Software available at http://www.csie.ntu.edu.tw/~cjlin/libsvm/

[12] Wu, C.L., Fu, LC., and Lian, F.L., "WLAN location determination in e-home via support vector classification", IEEE Int. Conf., 2004

[13] Correa, J., Katz, E., Collins, P., and Griss, M., "Room-Level Wi-Fi Location Tracking", Carnegie Mellon Silicon Valley, CyLab Mobility Research Center technical report MRC-TR-2008-02, Nov. 2008.

[14] Bolliger, P., Redpin, 2008. Software available at http://www.redpin.org/