Selective Mixture of Gaussians Clustering for Location Fingerprinting

Khuong An Nguyen
Computer Science Department
Royal Holloway, University of London
United Kingdom
khuong@cantab.net

Zhiyuan Luo
Computer Science Department
Royal Holloway, University of London
United Kingdom
zhiyuan@cs.rhul.ac.uk

ABSTRACT

One of the challenges of location fingerprinting to be deployed in the real offices is the training database handling process, which does not scale well with increasing amount of tracking space to be covered. However, little attention was paid to tackle such issue, where the majority of previous work rather focused on improving the tracking accuracy. In this paper, we propose a novel idea to enhance fingerprinting's processing speed and positioning accuracy with mixture of Gaussians clustering. We realised the key difference between fingerprinting and other un-supervised problems, that is we do know the label (the Cartesian co-ordinate) of the signal data in advance. This key information was largely ignored in previous work, where the fingerprinting clustering was based solely on the signal data information. By exploiting this information, we tackle the indoor signal multipath and shadowing with two-level signal data clustering and Cartesian co-ordinate clustering. We tested our approach in a real office environment with harsh indoor condition, and concluded that our clustering scheme does not only reduce the fingerprinting processing time, but also improves the positioning accuracy.

Keywords

Location fingerprinting, clustering, mixture of Gaussians.

1. INTRODUCTION

Indoor localisation is the state-of-the-art to monitor the position of a person inside a building, without the need of GPS coverage. In the past decade, location fingerprinting has been widely considered as one of the most effective indoor tracking methods to date. Fingerprinting-based approaches use the communication layers such as WLAN, Bluetooth, GSM and take advantage of the existing infrastructure to provide location tracking service. However, one of the challenges of fingerprinting to be deployed in the real offices is the training database handling process, which does not scale well with increasing amount of tracking space to be covered. In addition, multiple inquiries from many users need to be executed simultaneously on the training data, especially when a moving user needs to update his position continuously. Hence, this is undoubtedly the most costly and time consuming process of fingerprinting. However, little attention was paid to tackle such issue. Some researchers attempted to apply clustering - an unsupervised machine learning technique, to handle the fingerprinting database. However, due to the harsh indoor environment, real-time signal data may be allocated into the wrong cluster, resulting in poor tracking accuracy, a typical challenge of clustering for indoor localisation.

In this paper, we propose a novel idea to enhance fingerprinting's processing speed and positioning accuracy with mixture of Gaussians (MoG) clustering. We aim to tackle the indoor signal multipath and shadowing with signal data clustering and Cartesian co-ordinate clustering. Given a new sample, our approach picks a set of clusters based on their probabilities, while many previous work favours a single best cluster only. Since the design of our system is modular, we offer a more flexible approach to indoor fingerprinting clustering. For future research, we include our full fingerprinting database, which contains a floor plan, and all raw information such as Received Signal Strength, Link Quality, signal orientation, channel information, and more.

The paper begins with the challenges and current state-of-the-art of fingerprinting. We then explain our four steps to cluster the fingerprinting database. Empirical studies in a real office environment are discussed. Finally, we conclude our findings and outline the future work.

2. OVERVIEW OF LOCATION FINGERPRINTING

2.1 Current State-of-the-art and Challenges

Global Navigation Satellite Systems (GNSS) such as GPS are indispensable for outdoor navigation. However, people spend most of their times indoor, where limited or no GNSS service is available. The demands from big organisations such as supermarket and hospital to provide indoor navigation service to their customers and staff have encouraged much interest in the indoor localisation research in the past decade. Fine-grained indoor positioning systems with centimetre accuracy to coarse-grained room-level systems have been successfully reported [18, 21].

Since introduced in 2001, location fingerprinting has gained

I-LOCATE 2014, December 02-04, London, Great Britain

DOI 10.4108/icst.mobiquitous.2014.257983

much popularity due to its simplicity, which takes advantage of the existing building communication infrastructure such as WLAN [3] or Bluetooth [13]. The algorithm works with any type of signal that has locally constant power level, such as WLAN, Bluetooth, FM, GSM and 3G. WLAN Received Signal Strength Indicator (RSSI) was often used in the past literature to represent the indoor locations due to its ubiquity and simplicity. The fingerprinting method has two stages. In the first stage, which is known as the off-line phase, a training database pre-surveys the signals at every location in the building. In the on-line stage, which is known as the positioning phase, when a person wishes to discover his position, he measures the signals at his current location, and uses the training database to estimate a closest match. Fingerprinting can be viewed as a typical machine learning problem, where the training database composes of examples mapping the WLAN signal (the object), to its Cartesian x, y, z co-ordinate (the label). Our task is to predict the right label for a known object.

Modern commercialised solutions such as SkyHook[1], Ekahau[2], or free ones like Nokia's HERE[3] and Google Indoor Maps[4] all use fingerprinting to power their services. Although we do not know the exact details of the algorithms, the concept of fingerprinting still requires them to maintain a long list of indoor databases for each location or building floor. This raises a question if fingerprinting is the right direction for future indoor localisation? Below are some of our thoughts on the strengths and weaknesses of fingerprinting.

In terms of **accuracy**, fingerprinting is still a long way short of the extreme 3 cm achieved by those lateration and angulation-based systems [18] which use ultrasonic or pulse-width infrared signals to communicate between a wide range of fixed beacons and the user tags. Although we have seen a much improved sub-metre tracking accuracy reported in recent works with fingerprinting, typically with the use of Channel State Information CSI [5, 15, 20], there are multiple independent components such as the training data resolution, signal properties, and the algorithm itself, which all contribute to the end tracking result.

Availability can be the strength of fingerprinting, thanks to the ubiquitous indoor communication infrastructure such as WLAN or Bluetooth. Other long range outdoor signals such as FM, GSM, 3G can be used to boost low coverage indoor areas.

Installation and **ease of use** have their pros and cons. In most cases, the users only need to install an app on their mobile devices to enable tracking capability. Apart from a central server to exchange data with the users, no extra hardware is needed, because the whole idea takes advantage of the existing communication infrastructure of the building. However, the initial concept of fingerprinting does require an off-line site-survey phase, which adds burdens to the installation process.

[1]http://www.skyhookwireless.com
[2]http://www.ekahau.com
[3]http://here.com
[4]https://www.google.com/maps/about/partners/indoormaps

Maintenance is one of the weaknesses of fingerprinting. The training database becomes outdated over time, and to re-calibrate the whole tracking zone requires much labour work. This is one of the reasons why fingerprinting has yet been widely deployed in real offices.

Scalability is another major issue of fingerprinting. As the central server has to serve many users simultaneously, the processing speed does not scale well with the increasing number of users, especially when the training database is usually huge to cover a large tracking zone.

We have not discussed other aspects such as security, risk and reliability, since they are out of the scope of this paper. Clearly, one of the challenges for practical fingerprinting is how the training data should be handled. Apart from the maintenance issue, processing speed is a real concern, and will be tackled in this paper. Our work aims to reduce the computation overhead via a one-off clustering process to be performed when the fingerprinting database was first set up. In addition, we introduce our mathematical framework to select a set of clusters for a given new signal sample.

2.2 Related Work

Clustering techniques are diverse, and can be categorised into two broad groups. First, whether the clustering technique is hard clustering or soft clustering. The former partitions the dataset into disjoint groups (ie. k-Means (KM)), while the latter allows a data point to belong to more than one group (ie. MoG, fuzzy c-Means (FCM)). Second, whether the method is flat or hierarchical. Flat clustering such as KM generates a set of clusters with no explicit information to describe the relationship amongst those clusters. On the other hand, hierarchical clustering such as agglomerative uses a tree structure to describe the clusters.

Our clustering scheme combines soft clustering with a two-level tree structure. Our approach addresses the challenge of new signal sample standing in the borderline being misallocated to the wrong cluster, without having to explicitly generate overlapped cluster as proposed in [11]. For example, when a person stands near the wall of two rooms, which are assumed to belong to two separated clusters, he can be associated with either cluster. In [7, 10], the authors used Affinity Propagation to cluster the fingerprinting database, and only when the matching cluster has been identified based solely on the signal strength, then the Cartesian labels are used to output the user's location in the end. Similarly, [1, 17, 19] used KM and fuzzy logic to cluster the fingerprinting database based on the signal strength. With our approach, we combine the Cartesian label of each signal examples along with the signal data to tackle the signal multipath and shadowing problem. The authors tried to tackle a similar issue in [11], however, they proposed to drop the outliers all together, while our approach maintains the cluster. Our argument is if the user happens to stay in the position, where the signal training data has been removed, the system assumes he belongs to a different location with a similar signal reading.

3. FINGERPRINTING CLUSTERING

Clustering is an un-supervised learning problem, where the non-labelled data needs to be organised into groups with a

similar characteristic. With fingerprinting, the database is a mapping from signal data to Cartesian physical location. Since the task of fingerprinting is to find the Cartesian physical location for a user, given his current signal data, a popular choice in previous work was to cluster the fingerprinting database based on the signal data. The key difference between fingerprinting and other applications is that we do know the label (the Cartesian co-ordinate) of each signal data beforehand, where other datasets are normally non-labelled. However, this key information seems to be largely ignored when the clustering process takes place in previous work. In this section, we discuss the four modular steps of our clustering scheme. The performance of each step will be evaluated in the next section.

3.1 Finding Overlapped Clusters from Fingerprinting Database with Mixture of Gaussians

Learning mixture of Gaussians is a probabilistic model-based clustering, which uses the Gaussian distribution to model each cluster. We do not attempt to model the signal strength distribution, which may not be Gaussian at all. Our task is to find a model that best fit the entire fingerprinting database. To achieve that, we fit a Gaussian distribution model for each group of signal data from the fingerprinting database. The entire fingerprinting dataset is a sum or a mixture of these Gaussian distributions. In contrary to the wide belief that Gaussian distribution can only be used to represent normally distributed data, a mixture of a sufficient number of Gaussians can represent any data model.

The advantages of using MoG for fingerprinting clustering are the flexibility of allowing a signal data to be associated with more than one cluster, and the mixture model can represent the whole data well, if being fit properly. Compared to 'hard clustering' algorithms such as KM, where the training data is partitioned into disjoint clusters, our approach suits the fingerprinting problem better, since the observation points in the data are often cluttered together, resulting in multiple close clusters, in terms of their Cartesian or signal space. When a person stands at the borderline of two consecutive clusters, it may not be clear which cluster he belongs to with 'hard-clustering' algorithms. We chose the Gaussian distribution to model each cluster, because of its simplicity, and the ease to estimate the model parameters, to be discussed later.

We start with a multivariate Gaussian distribution function to calculate the probability that we have a signal strength vector $\mathbf{X} = (s_1, \ldots, s_n)$, with s_i is the RSSI observed from AP_i $(1 \leq i \leq n)$, given the mean vector μ and the covariance matrix $\mathbf{\Sigma}$ of a certain cluster.

$$p(\mathbf{X}|\mu, \mathbf{\Sigma}) = \frac{1}{(2\pi)^{\frac{n}{2}}|\mathbf{\Sigma}|^{\frac{1}{2}}} e^{-\frac{1}{2}(\mathbf{X}-\mu)^T \Sigma^{-1}(\mathbf{X}-\mu)} \quad . \qquad (1)$$

Let us assume our fingerprinting database $D = \{X_1, \ldots, X_m\}$ can be modelled by k Gaussian components (clusters). More details about choosing the value of k will be discussed later. Each Gaussian component $\mathcal{N}(\mu_i, \Sigma_i)$ is given a non-negative weight ω_i to represent the likelihood of occurrence of that component. These weights are important to help us decide which component (cluster) a new sample belongs to later on. For example, it is more likely that an un-observed sample belongs to the cluster with the highest weight. All these weights must sum up to 1 to maintain the probability distribution property of the model. The entire fingerprinting database's model is a mixture, or a weighted sum of these k clusters.

$$p(D|\omega, \mu, \mathbf{\Sigma}) = \omega_1 \mathcal{N}(D|\mu_\mathbf{1}, \mathbf{\Sigma_1}) + \cdots + \omega_k \mathcal{N}(D|\mu_\mathbf{k}, \mathbf{\Sigma_k}) \quad (2)$$

$$p(D|\omega, \mu, \mathbf{\Sigma}) = \sum_{i=1}^{k} \omega_i \mathcal{N}(D|\mu_\mathbf{i}, \mathbf{\Sigma_i}), \sum_{i=1}^{k} \omega_i = 1 \qquad (3)$$

Our objective is to maximise the above probability $p(D|\omega, \mu, \mathbf{\Sigma})$. In other words, we want to estimate the value of all the parameters ω_i, $\mu_\mathbf{i}$, and $\mathbf{\Sigma_i}$ $(1 \leq i \leq k)$ simultaneously to maximise the probability to observe the fingerprinting database D. A well-known solution to estimate said parameters is the Expectation-Maximisation (EM) algorithm [12]. This algorithm is particularly useful in our case, where the Gaussian mixture model we chose is fairly easy to maximise. We used the Gaussian Mixture Model package of Matlab[5] to perform the EM algorithm.

In summary, by the end of this step, we find a model which best fits the signal strength data from our fingerprinting database. Based on this model, we define a set of k overlapped clusters to group the signal strength vectors together. A vector may belongs to more than one cluster, and each cluster has a weight to determine its likelihood of occurrence in our probabilistic model. We will discuss the advantage of such weight and the selection of k later on.

3.2 Deriving Sub-clusters within a Cluster

One of the challenges for indoor localisation is the multipath and shadowing, caused by the reflection of the wireless signals from metal objects, diffraction around sharp corners, scattering off walls, floors and ceilings. This results in multiple copies of the original signals travelling in different directions. When two in-phase waves of a signal meet, constructive interference forms a new stronger wave of the signal. In contrast, two out-phase waves will cancel each other out, resulting in a weaker signal version. Therefore, two distinct locations in the building may have a similar signal strength observation. When such signal observations are put together in the same cluster based on their signal strengths, some of the signal data may not be near each other at all in their Cartesian space. Figure 1 demonstrates such phenomenon found in our office. With 7 clusters, there are two clear islands which form the black cluster, and three islands for the blue clusters. Although some of the minor blue members are found inside the cyan cluster, they do indeed possess similar signal data with the remaining members of their cluster. If a person happens to stay in the minority portions of this blue cluster, his location prediction result predicted by considering all members of the cluster will be pulled toward the majority of the blue cluster on the left. Our dataset and detailed experiments will be discussed later.

To tackle this issue, we perform the clustering process again

[5]http://www.mathworks.co.uk/help/stats/gmdistribution-class.html

Figure 1: Islands with similar WLAN signals visualised on 2-D floor plan of our office.

for each cluster found in the previous step. However, the members inside the cluster will be judged on their Cartesian x, y, z co-ordinate, rather than the signal strength vector as used in the previous step. Ideally, if all members within the cluster are close together in the Cartesian space, we will see one cluster after the process. Otherwise, multiple distinct clusters will be generated. Before we proceed any further, the following conditions should satisfy: (a) **The total number of clusters k is small**, in proportional to the size of the fingerprinting database. Since MoG already allows clusters to overlap, a high number of clusters may not benefit from having sub-clusters. Further, we will also pick more than one clusters for on-line positioning. (b) **The size of the cluster is large**. If a cluster is too small, it would not be beneficial to split it further. Without the need to repeat the clustering algorithm in the previous step, if the above conditions are met, we simply use the same algorithm again for individual cluster to separate their members into sub-clusters based on their Cartesian co-ordinate. These sub-clusters form a second layer in our two-level tree structure. The first layer is the original parents of these sub-clusters.

At the end of this step, we have got many (sub-) clusters, in which two of them may have the same signal characteristic. However, all of our clusters have different Cartesian characteristics. In the next part, we discuss our strategy to select a set of clusters, given the user's new signal data at his unknown location.

3.3 Selective Clusters for On-line Positioning

When a user wishes to discover his current position, he uses his mobile device to measure the signal data C at his location, and submits it to a central server. Then, the system identifies which clusters this signal data belongs to. This process is the most challenging and error-prone one for many clustering algorithms. If the new signal data is put into the wrong cluster, the location prediction accuracy will degrade as a consequence. While many previous work picked only the best matched cluster, our idea is to select a group of clusters instead. Since we already constructed a probability model of the fingerprinting database, we can calculate the probability of any new signal data to be associated to each of the k cluster, by substituting the new signal data C and two cluster's parameters $\mu_\mathbf{i}, \mathbf{\Sigma_i}$ $(1 \leq i \leq k)$ into the Gaussian Equation (4). Although the weight ω does not show in the equation, it influences the estimation of the parameters $\mu, \mathbf{\Sigma}$ in the model.

$$p(C|\mu_\mathbf{i}, \mathbf{\Sigma_i}) = \frac{1}{(2\pi)^{\frac{n}{2}} |\mathbf{\Sigma}|^{\frac{1}{2}}} e^{-\frac{1}{2}(C-\mu_\mathbf{i})^T \mathbf{\Sigma_i}^{-1}(C-\mu_\mathbf{i})} \quad . \tag{4}$$

By definition, these k probabilities will sum up to 1. Therefore, we can specify a threshold (ie. 90%) and pick all clusters with the largest probability adding up to the threshold. For example, with $k = 3$, and the three probabilities are $0.0, 0.2, 0.8$, we pick both the second and third clusters, for a 90% threshold. A high threshold will require more clusters to satisfy, while a smaller one needs fewer clusters. Without loss of generality, given a threshold value θ, $(0 \leq \theta \leq 1)$ and a decreasing order vector of k probabilities $P_k = (P_k^1, \ldots, P_k^k)$, we find a set of t probabilities $(t \leq k)$ to minimise the below equation.

$$arg \min_t [\sum_{i=1}^{t} P_k(i) \geq \theta] \quad . \tag{5}$$

In the next section, we learn how to estimate the user's position, given the set of clusters found in this step.

3.4 Finding the User Position

At this stage, a set of clusters has been identified given the user's signal data. We can treat this set of chosen clusters as our new, smaller fingerprinting database and apply our favourite algorithm to estimate the user's position, such as Naive Bayes or nearest neighbours. Since the main purpose of the paper is to accelerate the fingerprinting processing speed, and to reduce the search space via clustering, we leave out this section due to space limit. As our ultimate goal is to allocate the user's new data to the correct clusters, we will consider every member of the clusters to evaluate the positioning accuracy in this paper.

3.5 Tuning the Value of k

Choosing the optimal value for k can be as challenging as finding the clusters. There is still no unequivocal solution to find the optimal number of clusters. It is worth noting that bigger value of k does not necessarily mean longer running time, a criterion to judge the clustering algorithm to be evaluated later. Thus, the criteria for choosing k should be based on the structure of the training database, which is unfortunately different from every training set. The clustering scheme in this paper was designed with a relatively small number of clusters k in mind. If the reader chooses a big k, the cluster's size will be small, and deriving sub-clusters will probably not benefit the overall performance. Since the clustering process needs to be done just once, it is acceptable to combine a trial-and-error approach in which all the possible values of k are tested, with manual decision based on the result from each k. On the other hand, a well-known

practice to choose the value for k without manual decision is based on information criterion such as AIC (Akaike information criterion) and BIC (Bayesian information criterion), which are popular to evaluate model-based clustering such as MoG [8]. These information criteria measure the information loss corresponding to the model being used. The model with the smallest information criterion (AIC or BIC) is often selected. However, previous researches inclined to favour BIC for model-based clustering, based on their theoretical and empirical studies [6, 9, 14]. We will compare the performance of both AIC and BIC with our fingerprinting dataset in this paper.

3.6 Bringing It All Together

Figure 2 depicts our complete clustering scheme. Given a fingerprinting database mapping signal data to Cartesian co-ordinate, we use MoG to find k clusters based on the signal data only. Next, the MoG algorithm is applied again for individual cluster with the Cartesian co-ordinate as the clustering criteria, if the number of clusters in the previous step is small, and the size of the cluster is large. By the end of these two steps, we obtain a probability model for the entire fingerprinting database, and a small probability model for each cluster (if applicable). When a user submits his signal data to discover his current location, the system uses these models to highlight the clusters that best fit the user's signal data. Finally, the members inside the chosen clusters are compared directly with the user data to estimate the user's position.

4. EMPIRICAL EXPERIMENTS

4.1 Testbed

We will evaluate the performance of our clustering scheme in our office building, with a *48.1 m* x *45.7 m* floor plan. There are *9* WLAN APs directly inside the building (Figure 3). The WLAN adapter used to collect the signals are *Atheros AR928X WiFi 802.11b/g/n* integrated in our netbook.

Our dataset has an emphasis on the signal variation by capturing the signal repeatedly at each location. Such high level of signal density is particularly useful for the readers who wish to use our dataset with probabilistic approach experiments. There are *6,600* examples in our dataset. *6,471* of these will be used as training data, and the remaining *129* are used as test data. The majority of our test points are picked to be difficult to analyse, i.e borderlines points and those with a high degree of signal similarity amongst others. The distance between two consecutive training positions is approximately *80 cm*. At each position, we recorded the signal data *200* times, in four orientations corresponding to the North / West / South / East. The recorded signal metrics are Received Signal Strength Indicator (RSSI) and Link Quality (LQ). We will use RSSI to evaluate our system in this paper.

We provide the dataset publicly on our website[6]. To help further research, we include all raw information such as Channel ID, MAC address, orientation (N/W/S/E), both RSSI and LQ. Different from other training sets, we also provide the full floor plan. Each data point is not just mapped to the traditional Cartesian co-ordinate, but is also linked to a label on the floor plan, such as door tag, room number. Therefore, many data points can be recognised by their geographical information.

[6]http://www.cs.rhul.ac.uk/∼wruf265/

Figure 3: The heatmap of our testbed, generated by Ekahau.

4.2 Evaluation Criteria

Using the above testbed, we aim to answer the the following questions.

1. **Can clustering increase the tracking accuracy?** Ideally, our target is to maintain the same positioning accuracy with or without clustering involved in. However, by pinpointing the most relevant areas of interest (clusters), we may avoid redundant signal data examples, and the final prediction result may improve.

2. **How much computation overhead our system can reduce?** There are two computational heavy processes for fingerprinting. The first one is generating clusters from the off-line training data, and the second one is to find the correct cluster(s) for on-line positioning. We will assess both of the processes.

We will compare our clustering scheme against KM and FCM. KM is a well-known hard clustering algorithm which partitions data into distinct clusters. Since our clustering scheme use the same value of k, it is natural to choose KM as a competitor. On the other hand, FCM clustering is another popular clustering algorithm to produce overlapped clusters, which is one of the criteria to evaluate our algorithm. Since the design of our system is modular, we will evaluate individual steps, and the result of the entire process at the end.

4.3 Generating Clusters from Fingerprinting Database Evaluation

Figure 4 visualises the progression of the clustering process in our office from 1 cluster to 10 clusters. The data points are clustered based on the RSSI, then plotted on a 2-D map. With KM clustering, each group of data is virtually separated, while MoG allows a much higher degree of freedom for

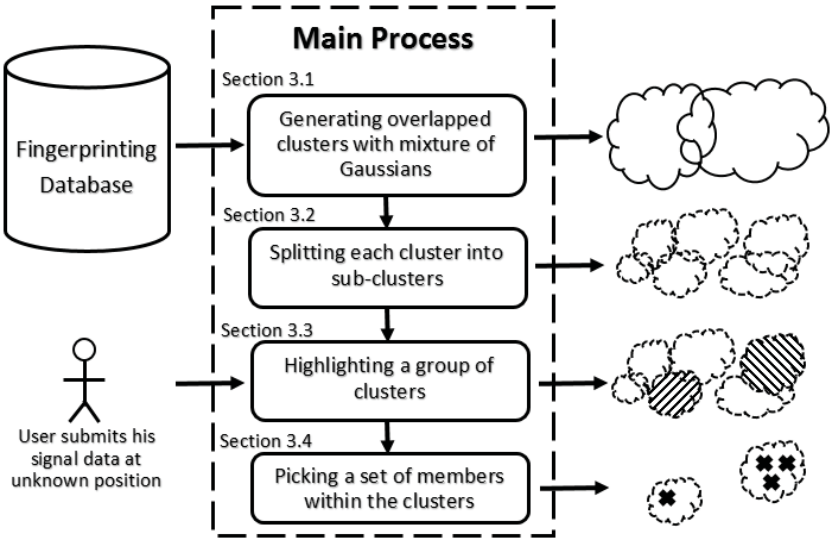


Figure 2: The progress of our clustering scheme.

overlapped clusters. Since each data point has a posterior probability corresponding to the likelihood of appearance for each cluster, the highest probability was used to decide which cluster the data point belongs to for MoG.

The main purpose at this step is to identify a near-optimal number of clusters k, based on AIC and BIC. Figure 5 suggests that the lowest value of BIC is around $k = 17$, while AIC shows a decreasing trend, where lower values are associated with much higher k, tested with all number of clusters from 1 to 50. The allocation and positioning accuracy (to be evaluated soon) proved that BIC was a better indication to select k with our fingerprinting dataset. Our result was similar to early theoretical reports that AIC tends to overestimate the number of clusters when used in mixture models [2, 4, 16].

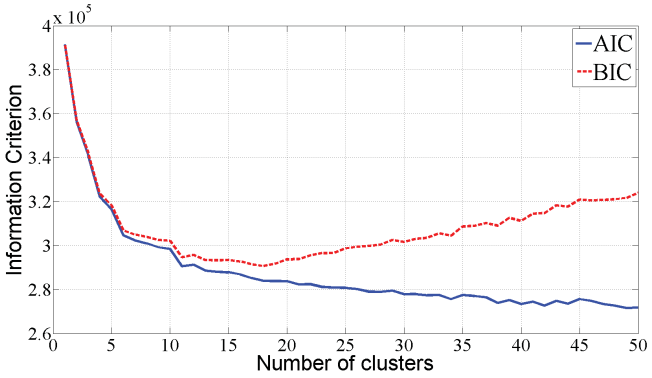


Figure 5: Optimal number of clusters with AIC and BIC.

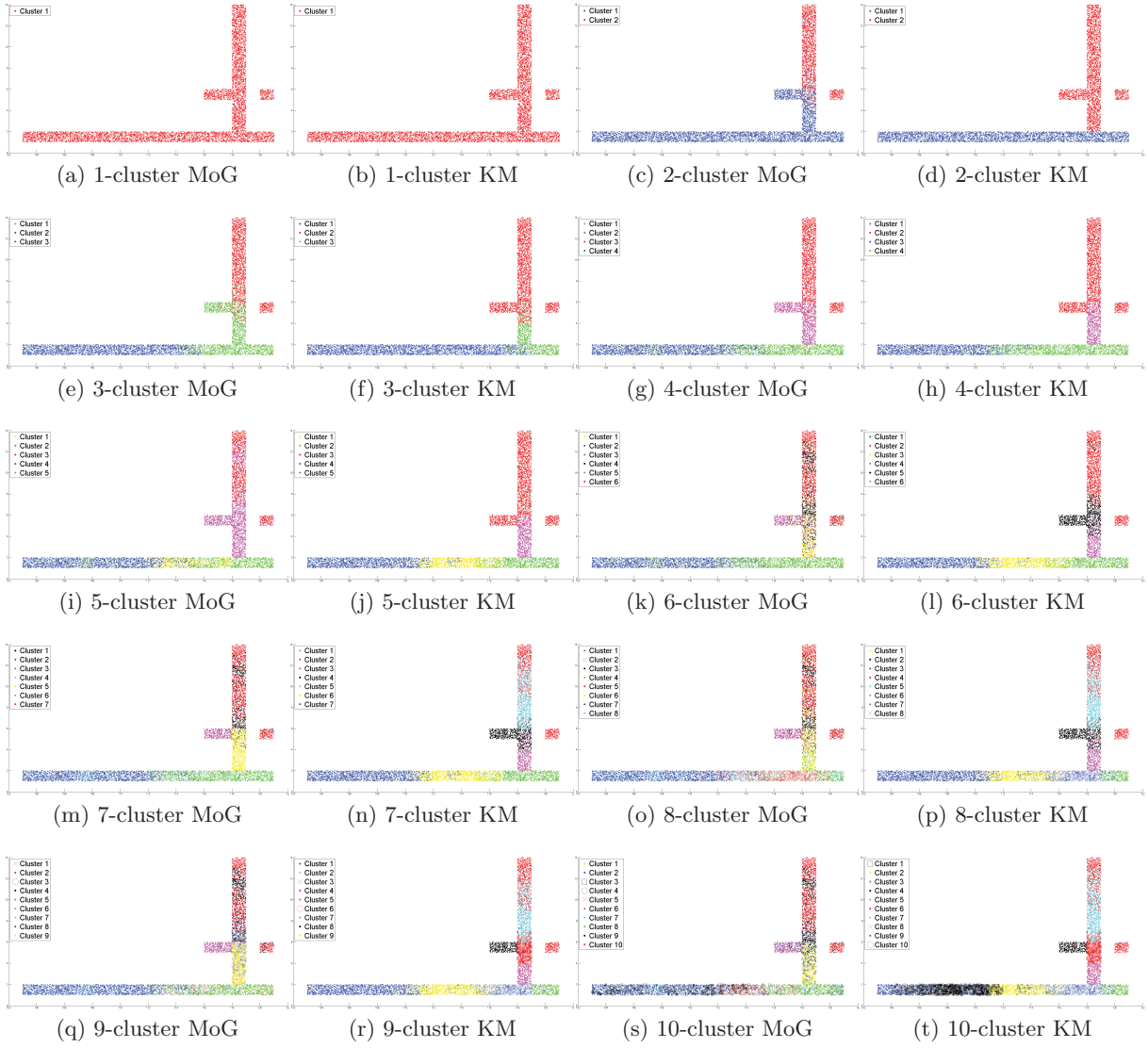
4.4 Deriving Sub-clusters Evaluation

As discussed earlier, this step is only useful if the total number of clusters is small, and the cluster size is big. Figure 6 demonstrates the size of individual cluster, as well as the mean and variation of a whole set of clusters generated by each $k = [1, 50]$. For each k, we ran the test 100 times with different initialisations, i.e different initial centroids for KM and different initial values for MoG. As the total number of clusters k increases, the box plot of MoG spreads out more, which shows a high degree of variation amongst the clusters' size. With MoG, we also noticed very big clusters, in proportion to the smaller size of other members, demonstrated by the red dots. On the other hand, KM maintains a steady size for its clusters, with very little variation. Combined with the result in the previous step, where the optimal k was suggested to be relatively low around $k = 17$, deriving sub-clusters may benefit the positioning accuracy, to be evaluated later.

Importantly, we noticed that FCM failed to produce the correct number of clusters, for certain values of k, such as 38,42,44,45,48,49 and 50. In our test, this happened more often with high dimensional vector or big k. Despite our effort in attempting different search iterations and improvement criteria, FCM did not manage to assign members to all required k clusters, resulting in several empty clusters. Previous work in FCM and fingerprinting did not report this issue with their dataset, although we notice that their datasets were much sparser and did not capture the full signal variation in different orientations as ours [17, 22].

4.5 Selective Clusters Evaluation

For a test sample $A = (RSSI_A, L_A)$, we used the Cartesian co-ordinate L_A to judge the decision making of the clustering algorithm. Normally, this Cartesian label is not available during the positioning stage, where only the signal information $RSSI_A$ is supplied from the user. The test sample A is decided to be correctly allocated to cluster X if the Euclidean distance between L_A and the average of all members of X is minimal. If there exists a cluster Y whose members are closer to L_A, we mark this sample's allocation as incorrect. It is worth noting that even if a test sample is flagged as incorrect allocation, it does not mean the positioning accuracy will suffer heavily, since the clusters generated by MoG are overlapped.



(a) 1-cluster MoG (b) 1-cluster KM (c) 2-cluster MoG (d) 2-cluster KM

(e) 3-cluster MoG (f) 3-cluster KM (g) 4-cluster MoG (h) 4-cluster KM

(i) 5-cluster MoG (j) 5-cluster KM (k) 6-cluster MoG (l) 6-cluster KM

(m) 7-cluster MoG (n) 7-cluster KM (o) 8-cluster MoG (p) 8-cluster KM

(q) 9-cluster MoG (r) 9-cluster KM (s) 10-cluster MoG (t) 10-cluster KM

Figure 4: Progress of WLAN signal strength clustering visualised on 2-D floor plan.

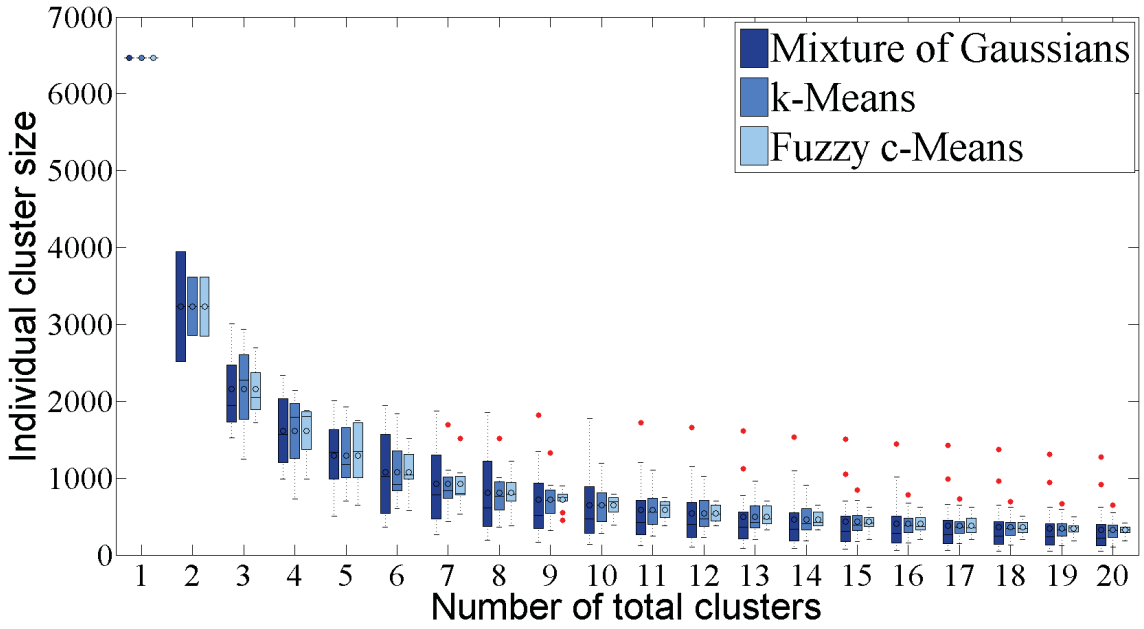


Figure 6: Individual cluster size, mean and variation.

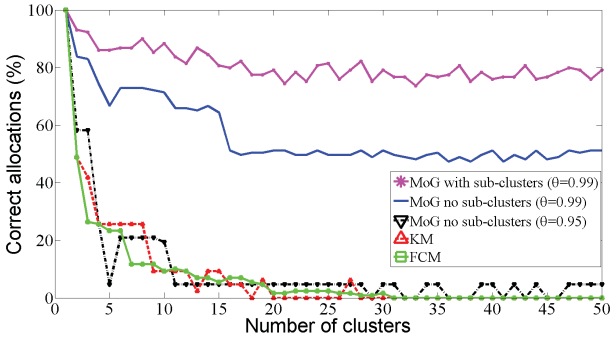


Figure 7: Number of correct allocations based on Cartesian label.

Figure 7 gives a good indication of what MoG achieved with or without sub-clusters, and with different values of threshold θ. For our dataset, threshold values $\theta \leq 0.95$ did not produce much impact on the number of selected clusters, which showed that the majority of the members had a strong confidence that they were in their correct cluster. We need to set the maximum value $\theta = 0.99$ for our dataset to achieve a higher rate of correct allocation with MoG. However, we spotted some exceptional signal data points being allocated to the wrong clusters, despite having a high probability. Upon close inspection, these data points did have a similar RSSI with other member of the same cluster, although they are positioned further away - the typical signal attenuation phenomenon. These exceptional test points were successfully put into their own clusters by applying our sub-clustering scheme. Overall, with MoG, we let just 20% of wrong allocations for the majority of k, while KM and FCM struggled from $k = 10$ with our dataset. In the next section, we will discuss how these wrongly allocated test points impact the positioning accuracy.

4.6 Final Location Prediction Evaluation

When a user submits his new signal data for positioning, we are interested in how accurate our proposed solution is, in terms of the distance (i.e metre) between the user's actual position and the estimated position. We compare three sets of performance in this section to highlight the benefit of using sub-clusters, the accuracy enhancement (if any) with/without clustering, and the positioning accuracy of MoG, KM and FCM.

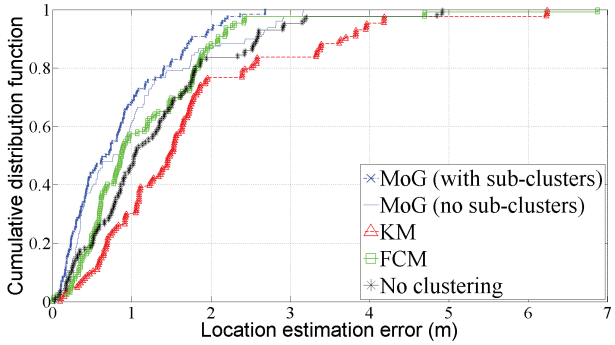


Figure 8: Location accuracy with k=17 clusters.

For simple evaluation purpose, we use no machine learning in this part. After the corresponding algorithm has identified the clusters which match the user's new signal data, we simply average all members of the clusters, and compare to the actual position recorded in the test data. The reader may treat these clusters as a new, smaller training data, and applies his favourite algorithm to choose a few best signal examples, rather than using the whole clusters.

At $k = 17$ (the optimal k as explained above), the estimation error was improved from less than 2 metres, 85% of the

time to less than 2 metre error for 90% of the time (Figure 8). Compared to KM, our approach and FCM produced better estimation at the same number of clusters, which confirmed the benefit of having overlapped clusters for fingerprinting. Finally, we employed a nearest neighbour approach with the entire fingerprinting dataset for non-clustering approach. Starting from 1-nearest neighbour to 500-nearest neighbours, we found the positioning accuracy of each case, and averaged results of all 500 cases for every test data. Compared to this non-clustering approach, MoG and FCM slightly reduced the estimation error. Hence, by pinpointing the most relevant clusters, we avoid redundant signal data examples and improve the performance accuracy as a result.

4.7 Processing Speed Evaluation

The training time, which the clusters are generated, is undoubtedly the most time-consuming process. The bigger the fingerprinting database, the longer it takes to generate clusters. Figure 9 shows that KM is the fastest option amongst the three algorithms to generate clusters, for any specified total number of cluster. All three algorithms used the same 200 iterations for a single running cycle, and 100 random cycles with different initial values (different initial centroids for KM) were repeated for each number of cluster to warrant data point convergence.

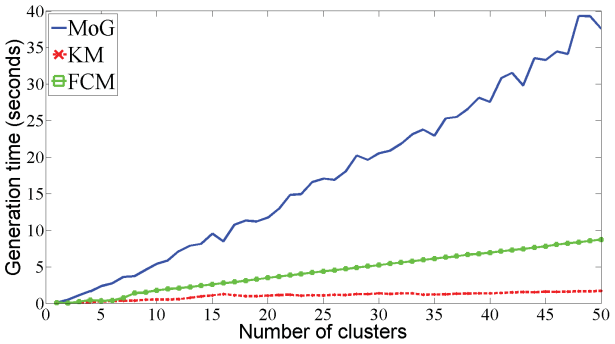


Figure 9: Average cluster generation time comparison.

However, when it comes to on-line positioning, which the user submits a new signal reading to discover his location, MoG performs as quick as KM as seen in Figure 10. This is expected, because we only need to use the new RSSI data as an input for the Gaussian equation, along with the mean vector μ and covariance matrix Σ found in our model, to calculate the probability of this new signal data for each of the k clusters. This is an advantage for MoG, considering its prediction accuracy is much better than KM and FCM as evaluated previously.

With non-clustering approach, every single training example must be searched through at least once for every test data. With our clustering approach for a total number of k clusters, firstly, we need to look through the representatives of k clusters once to choose a set of most relevant clusters for each test data. Then, we need to go through every element of this set to calculate the Cartesian position for the test data, which is similar to how non-clustering approach works. The smaller the value of k, the quicker the first process will finish, yet, the size of individual clusters will be large. Hence, the second process will run slower. In contrast, the bigger the value of k, the faster the second process finishes. Figure 11 shows that the total number of training examples we need to go through is almost 10 times smaller with our clustering approach (90% computational reduction) with all number of clusters from 1 to 50.

Figure 10: Average positioning time by searching through all clusters.

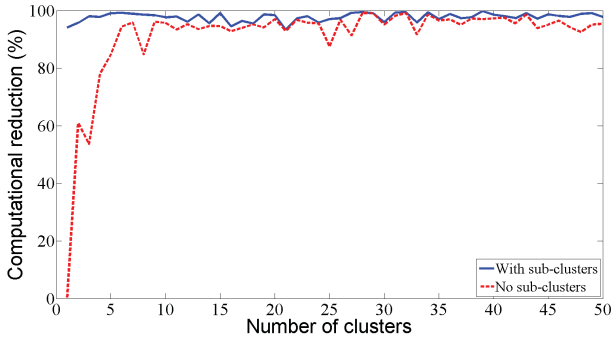


Figure 11: Computation overhead reduction via clustering.

5. CONCLUSIONS

We have demonstrated our idea to tackle the slow searching time during the on-line positioning phase of location fingerprinting with mixture of Gaussians clustering. With our approach, the signal data is allowed to be a member of more than one cluster. Our approach addresses the problem of a user stands at the borderline of multiple clusters, where the signal data may belongs to either cluster. Further, we realised the key difference between fingerprinting and other un-supervised problems, that is we do know the label (the Cartesian co-ordinate) of the signal data in advance. This key information was largely ignored in previous work, where the fingerprinting data was clustered based solely on the signal data information. By exploiting this information, we designed a two-level clustering process, where each cluster is further divided into sub-clusters based on their Cartesian label. Therefore, we tackled the typical signal attenuation phenomenon, which creates separated islands with similar signal in the same building. Finally, in contrary to the convention of choosing a single best cluster to represent the new signal data from the user, we made use of the probability information of each cluster offered by mixture of Gaussians to select a group of clusters for the on-line positioning phase.

We tested our solution in a real office building with harsh indoor environments. Based on the empirical results, we conclude that our approach effectively reduces the positioning time. More importantly, while KM does not perform well with borderline signal data, our approach maintains a low location positioning error, which was even slightly better than the performance of non-clustering algorithm, thanks to our attempt in pinpointing the most relevant areas of interest (clusters).

Our future work is to integrate our clustering scheme into a united fingerprinting system, which does not only tackle the burden in maintaining and updating the fingerprinting database, but also executes quickly in the real world with huge training data. We plan to deploy our system in a larger setting with the emphasis inside the adjacent rooms. Although our clustering technique alleviates the computation overhead to process the off-line database, the scalability of the system still depends on the number of users operating at the same time. We have a novel idea of lifting this burden from the central server completely by providing the mobile user a portion of the training database, which he needs to work out his current position himself.

6. ACKNOWLEDGMENTS

This research is funded by EPSRC grant EP/K033344/1 (Mining the Network Behaviour of Bots), and Computer Science Department, Royal Holloway University of London.

7. REFERENCES

[1] B. Altintas and T. Serif. Improving rss-based indoor positioning algorithm via k-means clustering. In *Wireless Conference 2011-Sustainable Wireless Technologies (European Wireless), 11th European*, pages 1–5. VDE, 2011.

[2] R. L. Andrews and I. S. Currim. A comparison of segment retention criteria for finite mixture logit models. *Journal of Marketing Research*, 40(2):235–243, 2003.

[3] P. Bahl and V. N. Padmanabhan. Radar: An in-building rf-based user location and tracking system. In *INFOCOM 2000. Nineteenth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE*, volume 2, pages 775–784. Ieee, 2000.

[4] G. Celeux and G. Soromenho. An entropy criterion for assessing the number of clusters in a mixture model. *Journal of classification*, 13(2):195–212, 1996.

[5] R. Crepaldi, J. Lee, R. Etkin, S.-J. Lee, and R. Kravets. Csi-sf: Estimating wireless channel state using csi sampling & fusion. In *INFOCOM, 2012 Proceedings IEEE*, pages 154–162. IEEE, 2012.

[6] A. Dasgupta and A. E. Raftery. Detecting features in spatial point processes with clutter via model-based clustering. *Journal of the American Statistical Association*, 93(441):294–302, 1998.

[7] G. Ding, Z. Tan, J. Zhang, and L. Zhang. Fingerprinting localization based on affinity propagation clustering and artificial neural networks. In *Wireless Communications and Networking Conference (WCNC), 2013 IEEE*, pages 2317–2322. IEEE, 2013.

[8] B. Everitt, S. Landau, and M. Leese. *Cluster Analysis*. Wiley, 2011.

[9] C. Keribin. Consistent estimation of the order of mixture models. *Sankhyā: The Indian Journal of Statistics, Series A*, pages 49–66, 2000.

[10] K. Li, J. Bigham, L. Tokarchuk, and E. L. Bodanese. A probabilistic approach to outdoor localization using clustering and principal component transformations. In *Wireless Communications and Mobile Computing Conference (IWCMC), 2013 9th International*, pages 1418–1423. IEEE, 2013.

[11] J. Ma, X. Li, X. Tao, and J. Lu. Cluster filtered knn: A wlan-based indoor positioning scheme. In *World of Wireless, Mobile and Multimedia Networks, 2008. WoWMoM 2008. 2008 International Symposium on a*, pages 1–8. IEEE, 2008.

[12] T. K. Moon. The expectation-maximization algorithm. *Signal processing magazine, IEEE*, 13(6):47–60, 1996.

[13] K. Nguyen and Z. Luo. Evaluation of bluetooth properties for indoor localisation. In *Progress in Location-Based Services*, pages 127–149. Springer, 2013.

[14] K. Roeder and L. Wasserman. Practical bayesian density estimation using mixtures of normals. *Journal of the American Statistical Association*, 92(439):894–902, 1997.

[15] S. Sen, R. R. Choudhury, B. Radunovic, and T. Minka. Precise indoor localization using phy layer information. In *Proceedings of the 10th ACM Workshop on Hot Topics in Networks*, page 18. ACM, 2011.

[16] G. Soromenho. Comparing approaches for testing the number of components in a finite mixture model. *Computational Statistics*, 9(1):65–78, 1994.

[17] D. J. Suroso, P. Cherntanomwong, P. Sooraksa, and J.-i. Takada. Location fingerprint technique using fuzzy c-means clustering algorithm for indoor localization. In *TENCON 2011-2011 IEEE Region 10 Conference*, pages 88–92. IEEE, 2011.

[18] R. Want, A. Hopper, V. Falcao, and J. Gibbons. The active badge location system. *ACM Transactions on Information Systems (TOIS)*, 10(1):91–102, 1992.

[19] C. Wu, Z. Yang, Y. Liu, and W. Xi. Will: Wireless indoor localization without site survey. *Parallel and Distributed Systems, IEEE Transactions on*, 24(4):839–848, 2013.

[20] Z. Yang, Z. Zhou, and Y. Liu. From rssi to csi: Indoor localization via channel response. *ACM Computing Surveys (CSUR)*, 46(2):25, 2013.

[21] M. Youssef and A. Agrawala. The horus wlan location determination system. In *Proceedings of the 3rd international conference on Mobile systems, applications, and services*, pages 205–218. ACM, 2005.

[22] H. Zhou and N. N. Van. Indoor fingerprint localization based on fuzzy c-means clustering. In *Measuring Technology and Mechatronics Automation (ICMTMA), 2014 Sixth International Conference on*, pages 337–340. IEEE, 2014.