Practical Concerns of Implementing Machine Learning Algorithms for W-LAN Location Fingerprinting

Jörg Schäfer
Frankfurt University of Applied Sciences
Faculty of Computer Science and Engineering
Nibelungenplatz 1
D-60318 Frankfurt am Main
E-mail: jschaefer@fb2.fh-frankfurt.de

***Abstract*—In the past, fingerprinting algorithms have been suggested as a practical and cost-effective means for deploying localisation services. Previous research, however, often assumes an (idealised) laboratory environment rather than a realistic set-up. In our work we analyse challenges occurring from a university environment which is characterised by hundreds of access points deployed and by heterogeneous mobile handsets of unknown technical specifications and quality. Our main emphasis lies on classification results for room detection. We analyse the problems caused by the huge number of access points available and by the heterogenous handsets. We show that standard techniques well-known in machine learning such as feature selection and dimensionality reduction do work. We also provide evidence that pre-processing techniques suggested previously in a laboratory set-up do not improve accuracy.**

I. Introduction

In this paper we describe results in deploying a localisation service in a university setting based on a fingerprinting approach. Although fingerprinting has been proposed in the past as a low-cost and effective approach to localisation services as it requires no special hardware to deploy, the theoretical analysis and experimental investigation of indoor localisation techniques which use RSS (Received Signal Strength) fingerprinting usually ignore the difficult impact of real-life environments different from the idealised laboratory set-up. In this paper we address three of these effects. First, instead of dealing with only a few selected and controlled access points we have to deal with hundreds of them. Secondly, this implies that one has to consider the impact of non-visible access points. And last but not least, an effective localisation service at a university has to cope with a great variety of mobile handsets the technical specifications of which are not known and cannot be determined with sensible effort. Note, that this aggravates the second problem as different handsets see different access points due to different antenna or chip sensitivity. The total effect of high density of access points on localisation accuracy is not fully understood and investigated in the sequel.

A. *Overview of Localisation Techniques*

Indoor localisation based on WiFi has long been analysed and studied, for a general overview we refer to [1]. Techniques can make use of a theoretical propagation model or can be based on empirical signal strength only. Although a theoretical underpinning with a propagation model sounds attractive from first principles, in reality it is impracticable as in real indoor environments it is very difficult to understand the propagation model from a theoretical perspective and close to impossible to effectively measure all the relevant characteristics as this measurement process is associated with prohibitive costs. Thus, simple models based on empirical signal strength, i.e. fingerprinting, are usually preferred.

B. *Project MoCa*

In 2012 at Frankfurt University of Applied Sciences the project MoCa (Mobile Campus Applications) was launched. Its mission is to provide personalised, context-sensitive services to students (and university staff) based on individual and role-based requirements. As the current position provides important context information for many services such as lecture and seminar support including voting services or social network services such as "find-your-buddy" applications, a localisation service is a core component of the MoCa infrastructure. Due to resource constraints, one of the key requirements of the project is zero maintenance which tries to minimise any maintenance efforts. Henceforth, any time-consuming efforts such as e.g. calibration must be avoided. Rather, we try to make use of crowd-sourcing techniques to facilitate the data generation (training) process. For a comprehensive overview of the MoCa vision we refer to [2].

II. Experimental Setup

A. Procedure

Data was collected at three floors in a building of the department of computer science and engineering, including altogether nine rooms of typical seminar and lecture room size (from 38 m^2 to 89 m^2; the layout is available upon request). In each room on average 53 reference locations were evenly spaced in a cartesian grid for tachymetric measurement. At each position, fingerprinting data was collected from 118 access points – most of which are not mutually visible at any single location. The (raw) measurements were recorded by an application on each handset and send to a central server.

B. Handsets

Training data was recorded using three different handsets as depicted in table I to model (proxy) for the heterogeneous clients to be expected in a production system. We refer to data generated by these clients as fp1, fp2, and fp3 in the sequel.

OEM	Model	MAC-Address	Data
Sony	Xperia S	30:39:26:0a:d2:cc	fp1
HTC	One X	1c:b0:94:b5:26:ea	fp2
Samsung	N700	50:cc:f8:1e:0c:d1	fp3

TABLE I: Mobile Handsets

III. Algorithms

A. Data Cleansing and NAs

Before using any data for machine learning, cleansing procedures are necessary to remove e.g. outliers. Checking revealed one missing data point which could be interpolated from its neighbours. Afterwards, several data transformations were applied as described in the next section. As the data contains a large number of missing, i.e. NA values due to missing RSS values of non observable access points – a result of the low signal strength below perception threshold at any particular location – these NAs have to be replaced with proper default values prior to applying machine learning algorithms such as KNN, SVM or PCA. For the raw data, most devices deliver RSS values down to ≈ -100 dBm, thus it seems to be reasonable to replace NA values with a default of -100 dBm. However, we consider this replacement part of the training phase of the learning algorithm and henceforth experimentally picked the optimal value. Furthermore, one has to carefully adjust the value for the default as the data distribution changes (mean and minimum values). Also, it has to be considered whether to first replace the NAs and then apply the transformations or vice versa as the operations are not commutative.

B. Data Transformation

It is well known in machine learning that a careful feature construction and transformation is important for a good learning result. In the sequel we adopt the following notation. Let us denote a single fingerprint by $\mathbf{x} = (x_1, \ldots, x_n)$, where n denotes the number of access points. The total of m fingerprints is denoted as $\mathbf{X} = (x_{ij})$, $i \in 1, \ldots, m$, $j \in 1, \ldots, n$ using matrix notation. In the sequel, transformations $\Phi : \mathbf{x} \mapsto \mathbf{x}'$ are applied to the fingerprinting values $\mathbf{x}$. We follow the usual convention that features are denoted by column- and measurements are denoted by row-values. Thus, in the sequel *linear transformations* are *identified* with *matrix operations* $\mathbf{M}$ operating from the *right*, i.e. $\Phi_M(x) := \mathbf{x}^\top\mathbf{M}$. (As usual, $\mathbf{x}^\top$ or $\mathbf{M}^\top$ denote the transpose of a vector $\mathbf{x}$ or matrix $\mathbf{M}$ resp.)

The following transformation algorithms were applied and are described in the sequel in detail:

1) Calculating the difference between access points – inspired by [3], we tried the following algorithms:
   a) Φ_{ssdR}: Subtracting the average of all distances
   b) Φ_{fBBA}: Calculating the distance to the strongest access point
   c) Φ_{fBBC}: Calculating the distance to the strongest access point per cell
   d) Φ_{fps2}: To assess how much information is contained receiving the signal of an access point *at all*, the data was category encoded into 1 or 0 according to whether or not a signal was intercepted
2) Projecting all access point to smaller number of principal components, as suggested in [4].

Differential RSS: It was argued in [5] that taking logarithms of the received signal strength measured in dBm – a technique called "hyperbolic fingerprinting" – could improve the results of localisation algorithms in the case of heterogeneous devices as taking the logarithm would compensate for the different reception of heterogenous devices. However, as pointed out in [3], the algorithm proposed lacks a theoretical foundation. Alternatively, in [3] it was proposed to take differences of RSS values, so-called SSD, measured in dBm. This procedure has a theoretical underpinning if one assumes a log normal shadowing model [6] for the signal distribution.

The SSD vectors are computed as follows

$$s_{ijk} := x_{ij} - x_{ik}. \tag{1}$$

In [3] it was shown that these s_{ijk} are (in first order) device independent: Let the i'th fingerprints x_{ij} and x_{ik} denote the received signal strength at a given distance d_j^i and d_k^i from access points a_j and a_k. Then in [3] the following equation is proven assuming a log normal shadowing model:

$$s_{ijk} = 10\log\left(\frac{P_{ap_j}G_{ap_j}\lambda^2_{ap_j}L_k}{P_{ap_k}G_{ap_k}\lambda^2_{ap_k}L_j}\right) - 10\beta_j\log\left(\frac{d^i_j}{d^i_0}\right) - 10\beta_k\log\left(\frac{d^i_k}{d^i_0}\right) - \left[X^i_j - X^i_k\right]_{dBm}, \quad (2)$$

whereas d^i_0 denotes the i'th distance to an (arbitrarily chosen) reference access point, and X^i_j, X^i_k denote the i'th received power from access point a_j and a_k, resp. (P_{ap_j} is the transmitted power, G_{ap_j} the antenna gain, λ_{ap_j} the wavelength, and L_j the system loss factor for a_j). The crucial observation is that equation 2 is free from any device dependent parameters and henceforth more suitable to be used as the input of an classification or (regression) algorithm. The first term in equation 2 is the same for all access points if they are configured identically and are of the same model[1] and is independent of any influence caused by variations of the handhelds; for details we refer to [3].

As there exist $n(n-1)/2$ different combinations of access point values, taking differences of signal strength seems to blow up the input space dimension accordingly. However, those differences are not linearly independent. Indeed, taking differences *reduces* the dimension of the input space from n to $n-1$. However, it is unclear, which access point to choose as a "reference" access point. Furthermore, even more importantly in our set-up, we cannot and should not use the *same* access point as a reference access point, as we want to cover a large area (university campus) and henceforth a single access point is not even visible at all interesting areas. Henceforth, we have to pick a different strategy. As we shall see, the strategies can be fit into a common framework.

For the following exposition it is convenient to denote the matrix projection operator onto the k-th RSS value in matrix notation by $\mathbf{\Pi}_k$, i.e. $(\mathbf{\Pi}_k^\top \mathbf{x}) := (x_k, \ldots, x_k)$. We denote the identity matrix by $\mathbf{I}$, i.e. $\mathbf{I}_{ij} = \delta_{ij}, \forall i, j$, where δ_{ij} denotes the usual Kronecker delta. Note that the SSD vectors implicitly define a matrix $(\mathbf{P_k}^\top \mathbf{x})_{ij} := s_{ijk}$.

Now the following identities hold trivially:

$$(\mathbf{\Pi}_k)_{ij} = \delta_{ki} \qquad \forall k, i, j \quad (3)$$

$$\mathbf{\Pi}_k \mathbf{\Pi}_l = \mathbf{\Pi}_k \qquad \forall k, l \quad (4)$$

$$\mathbf{P}_k = \mathbf{I} - \mathbf{\Pi}_k \qquad \forall k \quad (5)$$

$$\mathbf{P}_k \mathbf{P}_l = \mathbf{P}_\mathbf{l} \qquad \forall k, l \quad (6)$$

It follows that taking the SSD values are simple (linear), but non-orthogonal projection operations $\mathbf{P}_k$ with a one-dimensional kernel – the latter can be easily seen by directly computing that the unit vectors $\mathbf{e}_i$ for $i \neq k$ and $\mathbf{e}_0 := (1, \ldots, 1)$ form a basis of eigenvectors to the eigenvalues 1 and 0 resp.

In our work, we propose the following algorithm to reduce the dimensions: As each individual difference s_{ijk} is independent from the device characteristics, clearly any linear combination is so, too. Henceforth, in the sequel the averaging procedure, $\Phi_{ssdR} : \mathbf{x} \mapsto \mathbf{x}'$, defined as follows (with $\mathbf{J}$ denoting the all-ones matrix i.e. $\mathbf{J}_{ij} = 1, \forall i, j$):

$$\Phi_{ssdR} = \Phi_{\mathbf{I} - \frac{1}{n}\mathbf{J}} \quad (7)$$

is also independent from the device characteristics.

Then one can compute that[2]

$$(\Phi_{ssdR}(\mathbf{x}))_{ij} = x_{ij} - \sum_{k=1}^{n} \frac{x_{ik}}{n} \quad (8)$$

$$= \frac{1}{n}\sum_{k=1}^{n} \mathbf{P}_k, \quad (9)$$

is true, where n denotes the number of non-NA observations (access points), i.e. NAs are ignored (stripped out) if any exist.

(Note, that from equation 9 one immediately sees that Φ_{ssdR} can be written as a linear combination of SSDs. This algorithm collapses all SSDs into one single value and reduces the dimension accordingly to the maximum number of independent differences, i.e. to $n-1$.)

From simple matrix algebra (using equation 6) it follows that Φ_{ssdR} is a projection operator. It can be easily computed that the vectors $\mathbf{v}_1 := (1, -1, \ldots, 0)$, $\mathbf{v}_2 := (1, 0, -1, \ldots, 0)$, $\ldots$, $\mathbf{v}_{n-1} := (1, \ldots, -1)$ and $\mathbf{v}_n := (1, \ldots, 1)$ form a basis of eigenvectors to the $n-1$ eigenvalues 1 and 0 resp. Thus, Φ_{ssdR} has rank $n-1$ (or co-dimension 1).

As $\mathbf{P} := \Phi_{ssdR}$ is symmetric, i.e. $\mathbf{P}^\top = \mathbf{P}$ (as can be directly seen from 7), the projection $\mathbf{P}$ is orthogonal, i.e. every vector $\mathbf{x}$ can be written as $\mathbf{x} = \mathbf{x}_P \oplus \mathbf{x}_{P^\perp}$ where $\mathbf{x}_P := \mathbf{P}\mathbf{x}$ denotes the projection onto the subspace $\mathbf{P}$ (with $\mathbf{x}_{P^\perp} \in \ker \mathbf{P}$). It follows trivially, that

$$\|\mathbf{x} - \mathbf{y}\|^2 = \|\mathbf{P}\mathbf{x} - \mathbf{P}\mathbf{y}\|^2 + \|\mathbf{x}_{P^\perp} - \mathbf{y}_{P^\perp}\|^2. \quad (10)$$

Henceforth, if two vectors are close to each other in feature space, they have to be close to each other in the transformed (Im $\mathbf{P}$) space, too. Note that distances are *smaller* in the transformed space, hence, if data is linearly separable after the transformation it had to be linearly separable also before[3]. Henceforth, in the context of machine learning classification, we can expect

[1] not untypical for university deployments

[2] Recall that linear transformations are identified with matrix operations from the *right*!

[3] Similar arguments can be made for $\mathbf{P}_k$, too.

an improvement in the classification results iff the one-dimensional kernel that is projected out by $\mathbf{P}$ contains insignificant information only (e.g. noise). We also should aspect that algorithms that "learn" the significant features automatically from the training data such as SVM shall not benefit from this transformation if enough training data is available as the minimising margin hyperplane is found irrespective of (i.e. ignoring) $\ker \mathbf{P}$. As we shall see later, this intuition is confirmed by our data.

We also take differences to the strongest access point in the fingerprint, called Φ_{fBBA}, and defined as follows:

$$(\Phi_{fBBA}(\mathbf{x}))_{ij} := x_{ij} - \max_k \{x_{ik}\}, \tag{11}$$

In addition we compute Φ_{fBBC} as

$$(\Phi_{fBBC}(\mathbf{x}))_{ij} := x_{ij} - \max_{x_{kl} \in \mathrm{cell(x_{ij})}} \{x_{kl}\}, \tag{12}$$

where $x_{kl} \in \mathrm{cell(x_{ij})}$ implies that the fingerprint x_{kl} is taken from the same cell (room or sub-cell, see below) than x_{ik}.

Both algorithms also collapse all SSD into one value and reduce the dimension accordingly. They can be thought of picking the "right" access point from the SSD. As the picking strategy can be computed from the SSD data alone[4], it is clear that those transformations can be computed from the SSD values alone and henceforth are device independent (in first order). Note, that neither Φ_{fBBA} nor Φ_{fBBC} is linear. It should be noted that Φ_{fBBC} can only be used for training as in testing phase we do not know the correct location (cell). For testing, in this case, we proxy Φ_{fBBC} by the Φ_{fBBA} algorithm.

A preliminary analysis of the transformations based on collecting test-data was conducted in a bachelor thesis, see [7].

Binary RSS: The transformation Φ_{fps2} is defined as follows:

$$(\Phi_{fps2}(x))_{ij} := \begin{cases} 1 & \text{if } x_{ij} > -100 \\ 0 & \text{else.} \end{cases} \tag{13}$$

As we shall see below, this binary projection already contains a lot of location information.

PCA: Based on ideas in [4] a master thesis [8] was conducted to assess the applicability of principal component analysis at the university campus in the context of WLAN localisation. Principal component analysis (PCA) is a well-known statistical procedure that linearly transforms the coordinate system in such a way that possibly correlated variables are mapped into set of linearly uncorrelated variables called principal components. The number of principal components is less than or equal to the number of original variables. The eigenvalues corresponding to this transformation are sorted in descending order such that the first principal component has the largest possible variance, the second the second largest and so on. In the context of machine learning, principal components analysis can be used to reduce the number of observations and henceforth improve the signal to noise ration [9]. In our context, it is also important to note that the reduction of dimensions may lead to significant performance improvements which could be relevant for a production system.

[4]Because the definition of the max implies that we always pick the SSD with minimum total sum.

C. Performance Measures

For replacing NA values and tuning parameters of classification algorithms, the standard binary error measure e is used:

$$\mathrm{e}(\mathbf{x}, \mathbf{y}) := 1/n \sum_{i=1}^{n} \mathbf{1}(x_i \neq y_i), \tag{14}$$

where $\mathbf{x}$ and $\mathbf{y}$ denote predictions and measurements resp, n the number of observations and $\mathbf{1}$ the indicator function whose value is 1 if its argument is true and 0 otherwise.

For regression, the mean Euclidean distance r is used as well:

$$\mathrm{r}(\mathbf{x}, \mathbf{y}) = 1/n \sum_{i=1}^{n} \sqrt{|\mathbf{x} - \mathbf{y}|^2} \tag{15}$$

For classification, several error measures are known – for a systematic comparison, see [10]. To assess the performance of our algorithms, we calculate true positives tp, true negatives tn, false positives fp, and false negatives fn and compute precision $p := \frac{tp}{tp+fp}$ and *recall* (also called sensitivity) $r := \frac{tp}{tp+fn}$. Both measures are important in our context.

As precision and recall are conflicting goals, they can be combined into a so-called F-measure

$$f(\beta) := \frac{(\beta^2 + 1)\ tp}{(\beta^2 + 1)\ tp + \beta^2 fn + fp}, \tag{16}$$

where $\beta \in (0, \infty)$. The F-measure $f(2)$ is the harmonic mean of precision and recall. The closer the F-measure is to one the better is the classifier.

For the multi-class classification task, these binary performance measures can be combined into useful multi-class classification performance measures in several useful ways as explained in [10]. In our case we take a conservative approach and always choose the *minimum* (over all rooms, i.e. binary classification tasks) of either precision, recall or F-measure in the sequel.

D. Machine Learning Algorithms

KNN: We have chosen the well-known (deterministic) K-Nearest Neighbour algorithm KNN [11]. We have used the R-packages KKNN and FNN as implementations; the former [12] provides also support for different kernels and the latter is more performant (using e.g. cover trees as internal data structures). The latter also provides access to the indices of the nearest neighbours which makes it easy to calculate Euclidean distance errors, too.

SVM: Support Vector Machines (SVM) have risen in the past decade to become a powerful machine learning tool capable of representing non-linear relationships. In its simplest form a linear SVM finds the optimal linear separator, a hyperplane, between the two classes of data by maximising the so-called margin between the separating hyperplane and the data. If the data is non-separable some misclassified data points can be allowed and controlled by a regularising parameter. For truly non-linear problems, the data can be mapped to a – potentially high- (even infinite) dimensional feature space – by the so-called kernel-trick, for details see [13], [14], and [15].

Although SVMs can be used for regression as our main interest is in correct classification, we present results on using SVM mainly for classification tasks. As the data is (almost) linearly separable, we tried linear kernels[5] with good success. For our analysis we have chosen the R-package e1071[6].

Workflow: The subsequent results were computed based on the following workflow:

1) Apply data cleansing to raw data
2) Split data in training and test set by randomly splitting available data into training and validation data or validating against different test data generated by different device to assess impact of device heterogeneity
3) (Optional) NA replacement before applying the data transformations
4) (Optional) data transformation of either of Φ_{ssdR}, Φ_{fps2}, Φ_{fBBA}, or Φ_{fBBC}
5) If NA replacement has not occurred in step three, NA replacement
6) Computing optimal NA replacement values from training data by comparing cross validation results for nearest neighbour classifications
7) (Optional) apply principal components transformation (PCA)
8) (Re-) Training with the training set and using optimal Na values
9) Computing validation (cross validation)
10) Calculating all performance measures

IV. Results

The calculations based on the raw data[7] is analysed in the sequel. Selected results are shown in tables III and IV and figure 3.

[5]Experiments with non-linear such as Gaussian kernels did not show any improvements as was to be expected.

[6]which provides an interface to libsvm

[7]available upon request

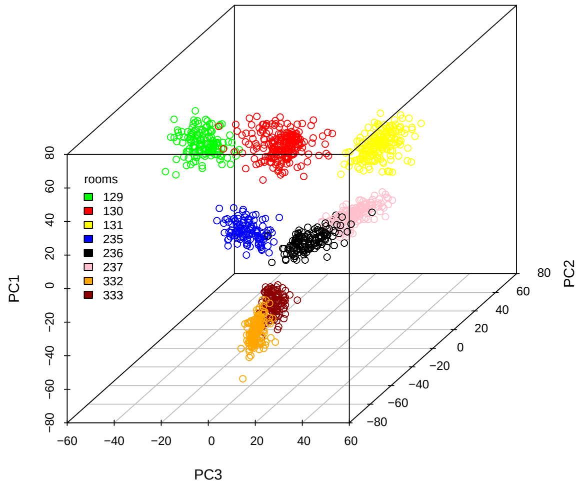

Fig. 1: Room Classification – PC1, PC2, and PC3

A. Geometric Structure of the Feature Space

Due to the high dimensionality of the feature space the data is linearly separable. This can be seen by performing a PCA and is confirmed using a SVM with linear kernel. As it turns out, the first two components already allow for linear separation of floor levels. Taking the first three components allows for almost linearly separating all rooms as illustrated in figure 1. This can also be deduced from computing the cumulative variances of the first principal components:

PC1	PC2	PC3	PC4	PC5	PC6	PC7	PC8	PC9
0.47	0.75	0.87	0.93	0.94	0.95	0.95	0.96	0.96

As one can see, the first six principal components add up to more than 95% variance.

The distribution of the pairwise (Euclidean) distance in feature space (see figure 2) demonstrates a high variability which implies that the data "lives" on a low-dimensional subspace of the n-dimensional feature space.

This can be concluded as well from calculating the correlation dimension $C(\epsilon)$ following [16] which is defined as the number of pairs with a distance below a threshold ϵ divided by the number of pairs as the number of observations tend to infinity:

$$C(\epsilon) := \lim_{m\to\infty} \frac{1}{m(m-1)} \#\{|x_i - x_j| < \epsilon\} \qquad (17)$$

The correlation dimension[8] is a measure of the dimensionality of the sub manifold the random data lives in, and does not have to be an integer.

[8]which was first introduced in fractal geometry

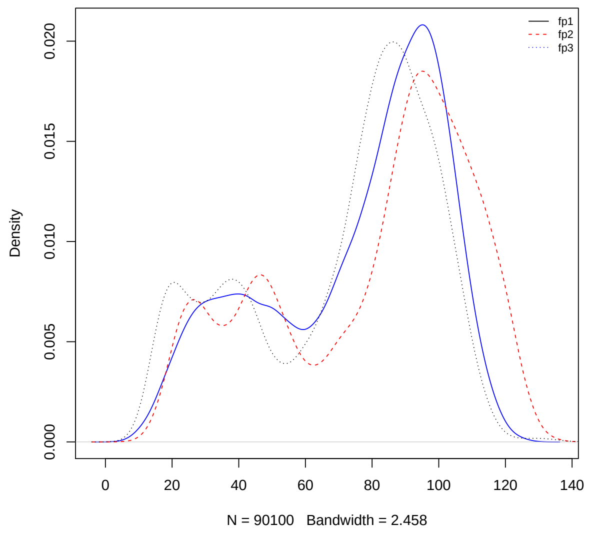

Fig. 2: Pairwise Distance Density

One estimates $C(\epsilon) \approx 3.1$ which is compatible with the PCA analysis and the fact that despite the high dimensionality of the feature space, KNN and other techniques based on geometric distance work well. Henceforth, the "curse of dimensionality" does not apply here.

B. Best NA Values

As typical devices do not record RSS below 100 considering equations 9, 11, and 12 we expect the following NA replacement values:

- For $\Phi_{identity}$ values around or below -100 dBm
- For Φ_{ssdR} values around or below -20 dBm
- For Φ_{fBBA} values around or below -60 dBm
- For Φ_{fBBC} values around or below -60 dBm

The empirically optimised values are close to these expected values.

C. Device Heterogeneity

The distribution (e.g. figure 2) also shows that the individual mobiles have quite different distributions of RSS values although the data was collected in exactly the same manner. Applying the Φ_{fBBA} transformation, the heterogeneity between the different handsets is reduced and the distributions look more uniform. However, as can be expected from the theoretical analysis above, this does *not* result in a better classification (nor regression) as we will see below.

Being able to use fingerprinting RSS for localisation by using machine learning algorithms that learn from calculating distances in feature space is possible, if and only if the distance in feature space[9] is correlated to the spatial distance. For indoor WLAN fingerprinting, however, this correlation is quite weak – a calculation reveals that the correlation between spatial pairwise distances d_s and pairwise distances in feature space d_f is only $corr(d_s, d_f) = 0.336$ for raw data (and only marginally improved if data is transformed by Φ_{ssd}). If instead Φ_{fBBA} is used, the correlation even *drops* dramatically to $corr(\Phi_{fBBA}(d_s), \Phi_{fBBA}(d_f)) = 0.111$.

For the identity transformation and the Φ_{ssd} and Φ_{fBBA} transformations we calculated the mean distance error distribution[10] applying the NA replacement before or after the transformations and could not find any significant difference.

Note, that in [3] SSD were investigated in a controlled lab set-up with only a few visible access points and only a few rooms. We therefore investigated whether SSD transformations yield improvements in mean distance error if applied in a more restricted setting. To this end, we filtered only measurements for each single room and applied the transformations accordingly. As one can deduce from table II, no systematic improvement occurs nonetheless.

rooms.id.	fps1	fps2	id	ssd	ap
235	fp1u2	fp3	2.54	2.37	2.28
235	fp1u3	fp2	2.69	2.42	2.58
235	fp2u3	fp1	2.75	2.39	2.38
236	fp1u2	fp3	2.15	2.35	2.51
236	fp1u3	fp2	2.72	2.73	2.83
236	fp2u3	fp1	2.82	2.73	2.82
237	fp1u2	fp3	2.45	2.58	2.49
237	fp1u3	fp2	2.48	2.40	2.40
237	fp2u3	fp1	2.68	2.63	2.56
333	fp1u2	fp3	3.67	3.63	4.06
333	fp1u3	fp2	3.61	3.86	4.02
333	fp2u3	fp1	4.07	3.53	3.87
332	fp1u2	fp3	2.56	2.73	2.40
332	fp1u3	fp2	2.22	2.33	2.59
332	fp2u3	fp1	2.95	2.46	2.81
129	fp1u2	fp3	3.25	3.09	3.23
129	fp1u3	fp2	3.09	3.05	2.97
129	fp2u3	fp1	3.15	3.51	3.20
130	fp1u2	fp3	3.20	3.76	3.47
130	fp1u3	fp2	3.26	3.20	3.49
130	fp2u3	fp1	3.60	3.20	3.39
131	fp1u2	fp3	3.03	3.18	2.75
131	fp1u3	fp2	3.12	3.07	3.13
131	fp2u3	fp1	3.17	2.98	3.26

TABLE II: Mean Distance Error per Room

It is only if one considers non-linear effects that SSD transformations make a difference. During the collection of measurements, by accident it was overlooked that the radio of device 1 does not record 5GHz, which resulted in artificially bad data quality. As this effect is non-linear and restricted to a few not spatially evenly distributed APs, applying SSD transformations indeed improved the errors *in this case* significantly.

A similar calculation confirms the analysis of the geometric features of differential RSS above. If one computes the pairwise distances in feature space of *projected eigenvectors*, the eigenvector to eigenvalue zero $\mathbf{v}_n = (1, \ldots, 1)$ has almost no correlation between spatial distances and distances in feature space, i.e. carries no

[9]usually induced by the standard Euclidean L^2 norm

[10]Taking the second and the third handset as training and the first as test data. The results for other combinations look similar.

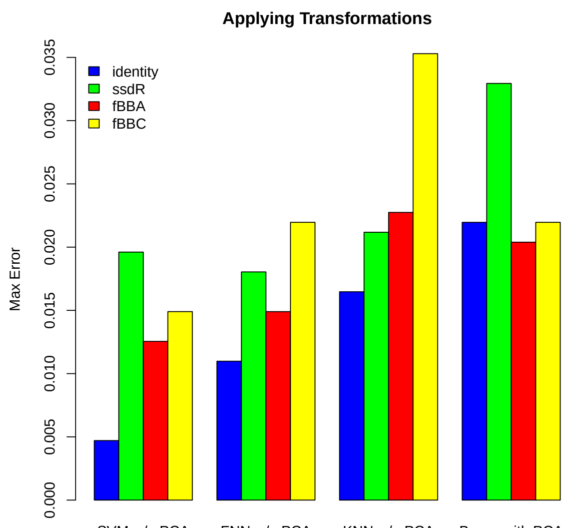

Fig. 3: Max Error - Two Mobiles vs. One

spatial information: $corr(\mathbf{P}_{\mathbf{v}_n}(d_s), \mathbf{P}_{\mathbf{v}_n}(d_f)) = 0.014$, whereas other eigenvectors have partially much higher correlation (up to ≈ 0.5). Henceforth, in the context of machine learning the spatial information contained in this particular dimension can be regarded as noise. This is not surprising as a good coverage of WLAN access points implies that every room has a similar average WLAN field strength, henceforth similar values in $\mathbf{v}_n$ direction. Thus, improving classification by taking differential RSS cannot be expected in this case.

D. Classification Results

Our main interest is classification of rooms in order to support context sensitive mobile application development. To this end, the machine learning algorithms were trained using the binary performance measure (see equation 14):

- Two vs. One Validation: Algorithms are trained with a combination of data from two of three mobiles and validated against the third mobile. For example data taken form the first and second mobile (fp1u2) is validated against the third mobile (fp3).
- All – Cross Validation: Algorithms are trained with a combination of data from all three mobiles. Data is randomly split into 10 folds to generate training and validation data.

As remarked, the transformations can be applied before or after NA replacement. In general, applying transformations before yields better results.

From the calculations we conclude the following findings:

	median(e)	max(e)	median(m)	min(p)	min(r)	min(f)
SVM w/o PCA	0.005	0.005	NA	0.971	0.978	0.985
FNN w/o PCA	0.011	0.011	3.076	0.954	0.954	0.975
KNN w/o PCA	0.016	0.016	NA	0.921	0.916	0.949
Bayes with PCA	0.022	0.022	NA	0.906	0.889	0.931

TABLE III: Performance Two vs. One Validation

	median(e)	max(e)	median(m)	min(p)	min(r)	min(f)
SVM w/o PCA	0.008	0.016	NA	0.857	0.846	0.917
FNN w/o PCA	0.004	0.024	2.773	0.909	0.846	0.917
Bayes with PCA	0.020	0.047	NA	0.833	0.727	0.842

TABLE IV: Performance All – Cross Validation

1) There is no significant difference in choosing different kernels in the KNN methods. However, optimising the kernel during training might lead to overfitting.
2) KNN (FNN) performs quite well on the raw data (i.e. only NA replacements and no transformation applied) despite the high dimensional feature space. This is probably due to data being concentrated on a low dimensional sub manifold such that arguments from e.g. [17] do not apply here.
3) SVM performs slightly better with carefully chosen parameters for linear kernels. The parameters were chosen by brute-force grid optimisation and are typically between $c = 0.01$ and $c = 10$ for the cost parameter c, $\gamma = 10^{-4}$ and $\gamma = 0.1$ for the gamma parameter γ with an average of 91 support vectors (out of 850 observations).
4) The Naive Bayesian approach fails because the assumptions of independence (and normal distributions) are violated.
5) However, if a PCA is performed, Bayesian performs quite well and is only slightly worse than SVM and KNN.
6) None of the data transformation procedures (except PCA for Bayes) improves results significantly and consistently as can be seen from figure 3.
7) The binary classification Φ_{fps2} already contains quite a bit of localisation information an yields a maximum error of $e \approx 0.09$.

Figure 3) shows that indeed transformations do not in general improve results. The best performing combinations SVM and FNN resp. KNN w/o PCA and Bayes with PCA are shown in tables III and IV.

As one can see, the leading algorithm SVM achieves almost perfect classification with a F-Measure $f = 0.985$ if trained with two mobiles (see table III) and an F-Measure of $f = 0.967$ if trained with all three mobiles[11] (see table IV). A typical – almost perfect – confusion matrix for a trained SVM is shown in table V.

KNN (FNN) is almost as good and the difference should not be regarded significant. For FNN we could compute mean distance errors around $r \approx 2.77$ meters.

[11]but using possibly less data in a particular room as data is randomly split

	129	130	131	235	236	237	332	333
129	48	1	0	0	0	0	0	0
130	0	70	0	0	0	0	0	0
131	0	0	63	0	0	0	0	0
235	0	0	0	40	0	0	0	0
236	0	0	0	0	47	1	0	0
237	0	0	0	0	0	52	0	0
332	0	0	0	0	0	0	53	0
333	0	0	0	0	0	0	0	50

TABLE V: Example of a Confusion Matrix

This is close to best results published in the literature [1] previously, if one considers that few outliers misclassified can result in large error values for the mean distance as our set-up is not a lab but a real building with measurements taken from huge distances.

Regression: We also calculated some regression results using KNN and SVM. To this end, regression were performed for each spatial coordinate x, y, and z individually rather than for the Euclidean distance directly[12]. Although this is not optimal, with the existing libraries it is easier to do. Not surprisingly, the regression results using KNN typically have errors compatible with the mean distance error calculated in the classification results. The regression results for SVM are worse and typically around 4.7 meters.

V. Conclusion

As we have shown, improving the results by preprocessing the data does neither lead to better results nor is necessary in an environment characterised by a high density of access points. The well-known machine learning algorithms SVM and KNN perform on par with SVM having a slight edge. Also a naive Bayesian approach can be used with good results if a principal component analysis is performed to transform the original problem into one that better fits the simplifying assumptions of the approach. The best algorithms to be used for an (almost) unsupervised maintenance free approach in a production environment still has to be determined. This is target of current investigations. To this end, crowdsourcing will be applied to facilitate the data collection process. This will include publishing open APIs both for data generation and service usage to encourage third-party application development on the mobile devices.

Another area of future research is combining the WLAN fingerprinting approach with the rich information from other sensors available in today's smartphone such as e.g. magnetic signatures, data taken from accelerometers and gyroscopes and ambient light.

Acknowledgment

We would like to thank Cornelia Eschelbach, Gerd Kehne, and Phillipp Winkelmann from Fb 1 for supporting the tachymetric measurements. We are grateful to Bishara Jaser who created the technology framework.

[12]This is related to multi task learning as the problem could be restated as learning the three dimensions altogether rather than individually, compare e.g. [18].

Note: This document was generated with Knitr using R version 3.0.2 (2013-09-25) on darwin10.8.0 and moca lib, V 0.4, depending on RMySQL 0.9.3, e1071 1.6.2, kknn 1.2.2, doMC 1.3.0, Hmisc 3.14.3, futile.logger 1.3.5, FNN 1.1.

References

[1] H. Liu, S. Member, H. Darabi, P. Banerjee, and J. Liu, "Survey of Wireless Indoor Positioning Techniques and Systems," vol. 37, no. 6, pp. 1067–1080, 2007.

[2] B. Jaser, "Design and set-up of an architecture for the development of a framework for location-based, mobile campus applications," Master Thesis, FH-Frankfurt, Jan. 2014.

[3] H. A.K.M. Mahtab, J. Yunye, H. Wee-Seng Soh, and V. Nguyen, "SSD: A Robust RF Location Fingerprint Addressing Mobile Devices' Heterogeneity," *IEEE Transactions on Mobile Computing*, vol. 12, no. 1, pp. 65–77, Jan. 2013. [Online]. Available: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6081874

[4] S.-h. Fang and T.-n. Lin, "Principal Component Localization in Indoor WLAN Environments," vol. 11, no. 1, pp. 100–110, 2012.

[5] M. B. Kj and C. V. Munk, "Hyperbolic Location Fingerprinting: A Calibration-Free Solution for Handling Differences in Signal Strength (concise contribution)," *2008 Sixth Annual IEEE International Conference on Pervasive Computing and Communications (PerCom)*, pp. 110–116, Mar. 2008. [Online]. Available: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=4517384

[6] T. S. Rappaport, *Wireless Communications: Principles and Practice*, 2nd ed. Prentice Hall, 2002.

[7] R. Killinger, "Implementierung von Algorithmen des Maschinellen Lernens in R," BA thesis, FH Frankfurt, 2013.

[8] I. Malinovskaya, "Mobile application providing location-based services for student community apps," Master's thesis, FH-Frankfurt, September 2012.

[9] C. M. Bishop, *Pattern Recognition and Machine Learning (Information Science and Statistics)*, 1st ed. Springer, 2007.

[10] M. Sokolova and G. Lapalme, "A systematic analysis of performance measures for classification tasks," *Information Processing & Management*, vol. 45, no. 4, pp. 427–437, Jul. 2009. [Online]. Available: http://linkinghub.elsevier.com/retrieve/pii/S0306457309000259

[11] E. Fix and J. L. Hodges, "Discriminatory analysis, nonparametric discrimination: Consistency properties," *US Air Force School of Aviation Medicine*, vol. Technical Report 4, no. 3, pp. 477+, Jan. 1951.

[12] K. Hechenbichler and K. Schliep, "Weighted k-nearest-neighbor techniques and ordinal classification," 2004. [Online]. Available: http://nbn-resolving.de/urn/resolver.pl?urn=nbn:de:bvb:19-epub-1769-9

[13] B. E. Boser, I. M. Guyon, and V. N. Vapnik, "A training algorithm for optimal margin classifiers," in *Proceedings of the Fifth Annual Workshop on Computational Learning Theory*, ser. COLT '92. New York, NY, USA: ACM, 1992, pp. 144–152. [Online]. Available: http://doi.acm.org/10.1145/130385.130401

[14] V. N. Vapnik, *The Nature of Statistical Learning Theory*. New York, NY, USA: Springer-Verlag New York, Inc., 1995.

[15] Vapnik, Vladimir N., *Statistical Learning Theory*. Wiley-Interscience, 1998.

[16] P. Grassberger and I. Procaccia, "Measuring the strangeness of strange attractors," *Physica D Nonlinear Phenomena*, vol. 9, pp. 189–208, Oct. 1983.

[17] K. Beyer, J. Goldstein, R. Ramakrishnan, and U. Shaft, "When is "nearest neighbor" meaningful?" in *In Int. Conf. on Database Theory*, 1999, pp. 217–235.

[18] L. Liang and V. Cherkassky, "Connection between SVM+ and multi-task learning," in *2008 IEEE International Joint Conference on Neural Networks (IEEE World Congress on Computational Intelligence)*. IEEE, Jun. 2008, pp. 2048–2054. [Online]. Available: http://dx.doi.org/10.1109/ijcnn.2008.4634079