JMLR: Workshop and Conference Proceedings 25:191–204, 2012 Asian Conference on Machine Learning

Frustratingly Simplified Deployment in WLAN Localization by Learning from Route Annotation

Ryoma Kawajiri KAWAJIRI@ICS.T.U-TOKYO.AC.JP
Masamichi Shimosaka SIMOSAKA@ICS.T.U-TOKYO.AC.JP
Rui Fukui FUKUI@ICS.T.U-TOKYO.AC.JP
Tomomasa Sato TSATO@ICS.T.U-TOKYO.AC.JP
Intelligent Cooperative Systems Laboratory, The University of Tokyo, 7-3-1 Hongo Bunkyo-ku Tokyo, 113-0033, Japan

Editor: Steven C.H. Hoi and Wray Buntine

Abstract

Recently wireless LAN (WLAN) localization systems are gaining popularity in pervasive computing, machine learning and sensor networks communities, especially indoor scenarios where GPS coverage is limited. To accurately predict location, a large amount of fingerprints composed of received signal strength values is necessary. Moreover, standard supervised or semi-supervised approaches also require location information to each fingerprint, where annotation work is rather tedious and time consuming. To reduce the efforts and time required to build calibration data, we present a novel calibration methodology "route-annotation" and a self-training algorithm for learning from route information effectively. On the proposed calibration methodology, an annotator walks around while measuring fingerprints, then occasionally stops to annotate fingerprints with route from previous location to current location. This calibration reduces work time even compared to partially annotation, while routes have richer information for learning. The proposed learning algorithm comprises following two iterative steps: 1) inferring locations of each fingerprint under route constraints and 2) updating parameters. Experimental results on real-world datasets demonstrate learning from route-annotated data is comparable to state-of-the-art supervised and semi-supervised approaches trained with large amount of calibration data.
Keywords: WLAN Localization, Route Annotation, Self-Training

1. Introduction

Recently wireless local area network (WLAN) localization systems are gaining popularity because of the widespread acceptance of mobile phones and WLAN access spots, especially in indoor scenarios where GPS is unavailable. In fact, several large-scale systems that can localize within 5-100 meters are realized for commercial service (Ekahau; Skyhook). Moreover, several researches using received signal strength (RSS) properties and statistical techniques show better accuracy within about 2-4 meters (Bahl et al., 2000). There is, however, difficulty to pursue both scalability and accuracy. One of the main reasons of the difficulty is that calibration work for whole buildings spends much time and effort. Before the system can be used, significant amount of fingerprints composed RSS values need to be measured in the whole deployment area. Moreover, it is also necessary to annotate every fingerprint with the locations in supervised approaches to achieve good accuracy on prior works (Bahl et al., 2000; Schwaighofer et al., 2003). The entire calibration process invariably requires human presence or other external localization system (e.g. GPS, camera arrays). As prior research, there are several semi-supervised approaches from partially labeled data for reducing annotating costs (Kashima et al., 2007; Pulkkinen et al., 2011). However, there remains big trade-off between accuracy and calibration effort. To tackle this problem, we present alternative simplified calibration methodology using "route-annotation", and a learning algorithm suitable for them.

To illustrate our idea with simple examples, comparison among typical calibration procedures and the proposed one is shown in Figure 1. Fully annotating locations is time consuming, since annotators have to wait for annotating current location and measuring fingerprints. As a partial solution to the problem, the annotators walk around while measuring fingerprints, then stops to annotate current location occasionally in order to evade spending useless time. This calibration method can reduce number of annotation and calibration time. However, unlabeled fingerprints have limited information for learning. Therefore, approaches with partially annotated data have low accuracy or still require significant amount of data. To overcome this dilemma, we propose a novel calibration methodology where annotators only need to annotate route from previous location to current location. While this annotation have essentially the same cost as partial annotation, routes have richer information compared with partial labels. Therefore, the amount of annotation can be drastically reduced.

Since correct locations are latent in route-annotated data, the standard supervised learning or semi-supervised learning algorithms (Ferris et al., 2006; Kashima et al., 2007) are not adaptable to route-annotated data directly. Hence, we introduce a variant of self-training algorithm (Rosenberg et al., 2005; Maeireizo et al., 2004) that can utilize route information effectively. This learning algorithm solves the non-convex problem by iterating two convex optimizations: latent locations inference and parameter update. In the inference step, locations of fingerprints measured are inferred quickly and stably, thanks to search space reduction by route information. In the update step, the method updates parameters by treating inferred locations as temporal supervisory signals.

The contributions of this paper are 1) proposal of calibration work using "route-annotation" that can incredibly reduce its cost and 2) development of learning framework that can appropriately treat route-annotated data. The rest of this paper is organized as follows. Section 2 formulates WLAN localization problems and compares our approach with related works. Section 3 presents the detailed description of the proposed learning algorithm. Section 4 shows the experimental evaluation on real-world datasets. Section 5 concludes our work.

2. Problem Setting and Related Works

This section first describes why this work focus on classification problem, then compares our approach to related works.

2.1. Problem Setting; localization as classification

Localization problems can be roughly categorized into *regression* approach and *classification* approach. In regression approach, the localization problem is defined as to predict locations $\boldsymbol{y} \in \mathbb{R}^2$ or $\mathbb{R}^3$ from fingerprints x. In classification approach, the localization

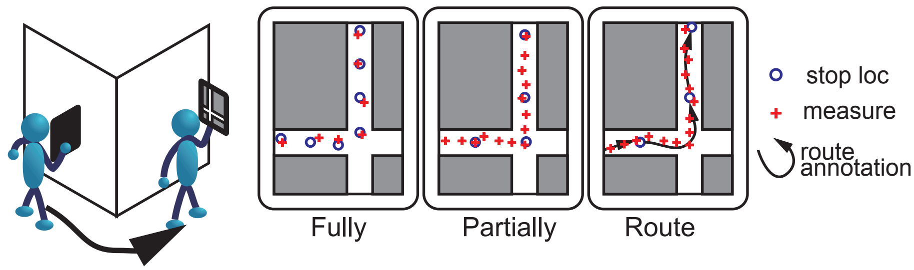

Figure 1: Comparison of calibration work by location and route

problem is defined as to predict location labels $y \in \mathcal{Y} = \{y_1, \ldots, y_N\}$. Many approaches on regression approach show good accuracy (2-4 meters) (Ferris et al., 2006; Pulkkinen et al., 2011), because the learning on regression approach is typically executed by minimizing distance error on dataset. However, typical floors in buildings, the main target of WLAN localization, are usually divided into several regions by walls, such as rooms or hallways. Classifying regions is still necessary before predicting precisely on regression approach. In the regression problem, the maps are represented with the combinations of region labels and region information such as region's shape and size. These complex maps can help to predict location accurately (Ferris et al., 2006), while it is hard to build complex large-scale maps. On the other hand, the representation on classification approach is only composed of location labels. Consequently it makes applicable to mature technology such as information retrieval method. In addition, loss metric with respect to error distance can be installed by using cost-sensitive learning on classification problem (Sen and Getoor, 2006). Actually, error needs to be considered by semantics rather than distance. For example in navigation situation on a crossroad, it is important to know on which road human is standing. For these reasons mentioned above, we focus on classification problems.

2.2. Related Works

There are several pioneering approaches from fully annotated data. First approach is k nearest neighbor (k-NN) method, proposed by Bahl and Padmanabhan (2000), that classifies objects by a majority vote of its neighbors, with the objects being assigned to the class most common among its k nearest neighbors. While k-NN method is easy to be implemented, it can take high accuracy (2-4 meters). Second, Schwaighofer et al. (2003) introduced localization method using Gaussian process. The approach can show good performance even on sparsely measured data. However, Gaussian Process methods have limitations in calculation resource and time, because memory space and computational time grow as $O(n^2)$ and $O(n^3)$ in simple implementation, where n is the number of training example. Furthermore, both of the above approaches require fully annotated data.

To reduce the cost of annotation, several semi-supervised approaches are studied. One of the prominent methods is label propagation method presented by Kashima et al. (2007) that wins on ICDM2007 data mining contest (Yang et al., 2008). Their approach defines similarity measures between examples, then build graph by connecting between similar examples, finally obtains labels of unlabeled examples by satisfying that labeled examples have the given labels and similar examples have the same location labels. Their approach, however, requires at least one labeled fingerprint for every label, even if there are many unlabeled fingerprints. In addition, their approach cannot completely utilize route-annotated data.

On natural language processing field, Fernandes and Brefeld (2011) reported a similar learning algorithm that transforms partially annotated data into fully annotated data. They, however, did not mention about the application of their idea to other fields or other annotation types. We demonstrate not only that similar learning algorithm can be applied to localization, but also that route annotation is more appropriate for localization.

There are other approaches to simplify calibration procedures. *Transfer Learning* is applied to WLAN localization by Pan et al. (2008); Zheng et al. (2008a,b). "Transfer learning aims to solve problem that the training data from source domain and the test data from a target domain follow different distributions or are represented in difference feature spaces" (Pan et al., 2008). The learning is applied to over time (Zheng et al., 2008b), across device (Zheng et al., 2008a). Even if transfer learning is used, it requires at least one well-annotated source domain data. Our approach helps reducing calibration cost even in source domain.

On the other approaches, the users of localization services collect calibration data instead of administrators of buildings. In these approaches, user-collected data are assumed as supplementary data to accurately predict or to correct errors due to time variance. However, to collect calibration data from the users, minimum localization systems need to be realized. Thus, the cost of initializing the system is still expensive. Our approach reduces initializing cost and can also learn from supplementary data.

3. Learning from Route Annotated Data

We now provide the detailed description of the learning method from route-annotated data. The WLAN localization problem can be defined as finding the most possible sequential locations $\boldsymbol{y} = y_{1:T}$ from all candidate locations $\mathcal{Y}$ when sequential fingerprints $\boldsymbol{x} = x_{1:T}$ are measured. T denotes the length of $\boldsymbol{y}$ and $\boldsymbol{x}$. This can be represented as follows:

$$\hat{\boldsymbol{y}} \quad = \quad \operatorname*{arg\,max}_{\boldsymbol{y}\in\mathcal{Y}} \; s(\boldsymbol{y}, \boldsymbol{x}; \boldsymbol{w}), \tag{1}$$

where $\hat{\boldsymbol{y}}$ denotes a predicted location, and $s(\boldsymbol{y}, \boldsymbol{x}; \boldsymbol{w})$ denotes a score function represents the possibility, and $\boldsymbol{w}$ is parameters for the score function. Standard supervised learning requires fully annotated dataset to determine parameters $\boldsymbol{w}$. A fully annotated dataset comprising N fingerprints $\boldsymbol{x}^{(1:N)}$ with corresponding location $\boldsymbol{y}^{(1:N)}$ is represented by $D_{full} = \{\boldsymbol{x}^{(n)}, \boldsymbol{y}^{(n)}\}_{n=1:N}$. With definition of a function l quantifying the loss for each set $(\boldsymbol{x}^{(n)}, \boldsymbol{y}^{(n)})$, the parameters $\boldsymbol{w}$ are obtained by minimizing empirical loss as following function:

$$\min_{\boldsymbol{w}} \sum_{n=1}^{N} l(\boldsymbol{y}^{(n)}, \boldsymbol{x}^{(n)}; \boldsymbol{w}). \tag{2}$$

Several loss functions l can be selected depending on problems, for instance, distances between location labels, slack variables on Max-Margin methods, etc.

Partially annotated or route-annotated data do not have the labels for every fingerprint $\boldsymbol{x}^{(n)}$. Therefore, instead of the ground-truth annotation $\boldsymbol{y}^{(n)}$, we introduce latent labels $\boldsymbol{z}^{(n)} = z^{(n)}_{1:T}$ by referring to (Felzenszwalb et al., 2008). The empirical loss function including latent labels can be defined in the similar way:

$$\min_{\boldsymbol{w}} \sum_{n=1}^{N} l(\boldsymbol{z}^{(n)}, \boldsymbol{x}^{(n)}; \boldsymbol{w}). \tag{3}$$

To minimize this function, we developed a form of self-training, which is a type of semi-supervised learning. The learning is executed by iteration of the following two steps: 1) inference of structured latent location labels from route and 2) updating parameters by using the inferred latent labels. Below sections describe the details of each step in the algorithm.

3.1. Inference of Latent Location Labels

This section describes the algorithm of inferring latent location labels from partially annotated and route annotated data, then compares them. A partially annotated data set comprising M time-series fingerprints $\boldsymbol{x}^{(m)} = x^{(m)}_{1:T_m}$ with corresponding partial labels $\boldsymbol{p}^{(m)} = \{t^{(m)}_j, y^{(m)}_j\}_{j=1:L_m}$ $(1 \leq t^{(m)}_j \leq T_m)$ can be represented as $D_{part} = \{\boldsymbol{x}^{(m)}, \boldsymbol{p}^{(m)}\}_{m=1:M}$. The constraints for inference steps on partially annotated data are 1) labeled data are kept their labels, 2) temporally adjacent labels are the same label or locally adjacent. When $\Gamma_p(y)$ denotes a set of label y and adjacent locations labels of y, the latent labels are obtained as blow

$$\begin{aligned}
\boldsymbol{z}^{(m)} = \; & \underset{\boldsymbol{z}^{(m)}}{\arg\max} & & s(\boldsymbol{z}^{(m)}, \boldsymbol{x}^{(m)}; \boldsymbol{w}) \\
& \text{subject to} & & z^{(m)}_{t_j} = y^{(m)}_j \; (j = 1, \cdots, L_m). \\
& & & z^{(m)}_{t+1} \in \Gamma_p(z^{(m)}_t) \; (t = 1, \ldots, T_m - 1).
\end{aligned} \tag{4}$$

A route-annotated dataset is similarly represented as $D_{route} = \{\boldsymbol{x}^{(m)}, \boldsymbol{r}^{(m)}\}_{m=1:M}$, where $\boldsymbol{r}^{(m)}$ denotes route information. In the classification problem, the route data are represented by series of labels ordered in which an annotator passed, as $\boldsymbol{r}^{(m)} = r^{(m)}_{1:L_m}$. The constrains are given as below: 1) the first and the last fingerprints are labeled as the start and the end labels of route respectively and 2) temporally adjacent labels are same labels or next label on the route. When $\Gamma_r(r_l)$ denotes a set of label y_l and a next label on route r_{l+1} $(\Gamma_r(r_l) = \{r_l, r_{l+1}\})$, the inference step on route data is represented as below:

$$\begin{aligned}
\boldsymbol{z}^{(m)} = \; & \underset{\boldsymbol{z}^{(m)}}{\arg\max} & & s(\boldsymbol{z}^{(m)}, \boldsymbol{x}^{(m)}; \boldsymbol{w}) \\
& \text{subject to} & & z^{(m)}_1 = r^{(m)}_1, z^{(m)}_{T_m} = r^{(m)}_{L_m}. \\
& & & z^{(m)}_{t+1} \in \Gamma_r(z^{(m)}_t) \; (t = 1, \ldots, T_m - 1).
\end{aligned} \tag{5}$$

Figure 2 shows comparisons between the two constraints of each inference. The route constraints drastically decrease candidates compared to the inference from partially annotated data. Consequently, the inference of route annotated can efficiently perform and stably derive optimum solutions.

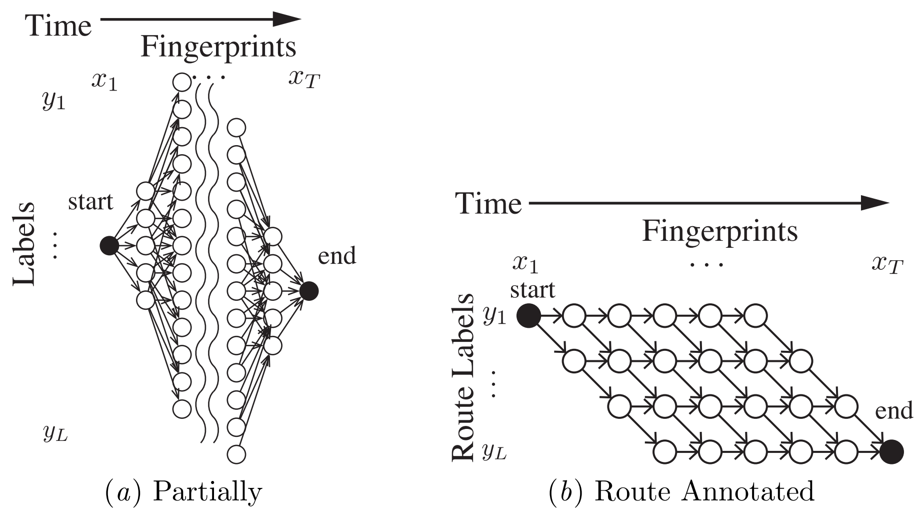

Figure 2: Comparison of constraints on partially labeled and route annotated data

3.2. Minimize Cost-sensitive Hinge-loss with Inferred Latent Variables

This section describes parameter update step that minimizes cost-sensitive hinge-loss by referring to SVM struct (Tsochantaridis et al., 2006). This method is appropriate for learning from inferred latent variables and for WLAN localization.

3.2.1. Cost Sensitive Max-Margin

In Max-Margin approaches, $s(y, x; \boldsymbol{w})$ is defined by taking a linear function of the feature vector $\boldsymbol{\phi}(y, x)$

$$s(y, x; \boldsymbol{w}) = \boldsymbol{w}^{\intercal}\boldsymbol{\phi}(y, x) \tag{6}$$

where $\boldsymbol{w}$ is a weight vector obtained via learning. As loss function l mentioned Equations (2) and (3), we introduce a cost-sensitive slack variable ξ. When $\Delta(y, \hat{y})$ is a cost function between the true label y and the prediction $\hat{y}$, slack variables $\xi(y, x)$ become as follows:

$$H(y, \hat{y}, x) = \Delta(y, \hat{y})\,(1 - s(y, x) + s(\hat{y}, x)) \tag{7}$$

$$\xi(y, x) = \max\left(0, \max_{\tilde{y} \in Y_{\setminus y}} H(y, \tilde{y}, x)\right) \tag{8}$$

To prevent overfit and sparsely select features, we use L_1 norm regularization. Finally, we get the object function as follows:

$$\min_{\boldsymbol{w}} \left(\sum_{n=1}^{N} \xi_n + \lambda\|\boldsymbol{w}\|_1\right), \tag{9}$$

where $\xi_n = \xi(y_n, x_n)$, and $\lambda > 0$ is a constant that controls the trade-off between training error minimization and regularization.

3.2.2. Learning Algorithm

We use stochastic gradient descent (SGD) as learning algorithm. The learning algorithm is executed by the iteration of the transformation steps and parameter update steps until the weight vectors $\boldsymbol{w}$ converge. In the transformation step, pseudo labels Z_m are obtained from route-annotated data $\{X_m, Y_m\}$ by solving Equation (5) (transform t). The weight parameters are then updated with subset of dataset $\{X_m, Z_m\}$ (update u). The whole learning is simply summarized as Algorithm 1.

Algorithm 1 Learning Algorithm

Require:
 Data Set: $D = \{X_m, Y_m\}_{m=1:M}$
 Cost Function: $\Delta(y, \hat{y})$
1: Variable:
 Weight vectors: $\boldsymbol{w}$
2: Initialization: $\boldsymbol{w} \leftarrow \mathbf{0}$
3: **repeat**
4: **for** $m = 1, \ldots, M$ **do**
5: $Z_m \leftarrow t(X_m, Y_m, W)$
6: $W \leftarrow u(W, Z_m, X_m, \Delta)$
7: **end for**
8: **until** $\boldsymbol{w}$ converge
9: **return** $\boldsymbol{w}$

As the update algorithm, we adapt a variant of FOBOS (Duchi and Singer, 2009), which is simple to implement and reported fast especially for L_1 norm regularization. For simplicity, we rename the subsets X_m, Z_m as $X_s = x_{1:N_s}$, $Y_s = y_{n=1:N_s}$. The update step is further divided into two steps as follows:

$$\boldsymbol{v} \leftarrow \boldsymbol{w} - \eta \sum_{n=1}^{N_s} \left(\frac{\partial \xi_n}{\partial \boldsymbol{w}} \right), \tag{10}$$

$$w_i \leftarrow \operatorname{sign}(v_i) \max \left\{ 0, |v_i| - N_s \eta_{\frac{1}{2}} \lambda \right\}. \tag{11}$$

where $\boldsymbol{v}$ denotes temporal weight vectors, $\eta, \eta_{\frac{1}{2}} > 0$ is learning rate, w_i, v_i denotes i th elements of weight vectors $\boldsymbol{w}, \boldsymbol{v}$. The gradient for each sample $\frac{\partial \xi_n}{\partial \boldsymbol{w}}$ can be obtained by

$$\frac{\partial \xi_n}{\partial \boldsymbol{w}} = -\Delta(y_n, \tilde{y}_n) \left(\boldsymbol{\phi}(y_n, x) - \boldsymbol{\phi}(\tilde{y}_n, x) \right) \tag{12}$$

where $\tilde{y}_n$ is the most violent label, $\tilde{y}_n$ is given by

$$\tilde{y}_n = \operatorname*{arg\,max}_{y \in Y_{\setminus y_n}} H(y_n, y, x_n). \tag{13}$$

We summarize the update algorithm on Algorithm 2.

Algorithm 2 Update Algorithm

Require:
Subset of data: $Y_s = \{y_n\}_{n=1:N_s}$, $X_s = \{x_n\}_{n=1:N_s}$
Weight vectors: $\boldsymbol{w}$

```
Initialization: g ← 0
/* calculate gradient on subset of dataset*/
for n = 1 : N_s do
   if ξ(y_n, x_n) > 0 then
      g ← g + η ∂ξ_n/∂w
   end if
end for
/* update weight vectors*/
Unconstrained gradient descent step: v ← w − ηg
Regularization Step: w_i ← sign(v_i) max{0, |v_i| − N_s η_{1/2} λ} (∀i)
return w
```

1: Initialization: $\boldsymbol{g} \leftarrow \mathbf{0}$
/* calculate gradient on subset of dataset*/
2: **for** $n = 1 : N_s$ **do**
3: **if** $\xi(y_n, x_n) > 0$ **then**
4: $\boldsymbol{g} \leftarrow \boldsymbol{g} + \eta \frac{\partial \xi_n}{\partial \boldsymbol{w}}$
5: **end if**
6: **end for**
/* update weight vectors*/
7: Unconstrained gradient descent step: $\boldsymbol{v} \leftarrow \boldsymbol{w} - \eta \boldsymbol{g}$
8: Regularization Step: $w_i \leftarrow \mathrm{sign}\,(v_i) \max\left\{0, |v_i| - N_s \eta_{\frac{1}{2}} \lambda\right\}$ $(\forall i)$
9: **return** $\boldsymbol{w}$

3.3. Localization

Estimating locations from time series fingerprints can show good performance by using prior location and velocity (Ferris et al., 2006; Bahl et al., 2000). However, successive scanning is not practical especially for mobile devices. Since, while scanning WLAN networks, data communication cannot be used and much battery resources are consumed. When localization system is used for indoor navigation, a user searches his current location when he lost. Successive estimation is, therefore, not always necessary. It is more important to localize accurately at once. Therefore, the actual localization can be simply represented as

$$\hat{y} = \underset{y \in \mathcal{Y}}{\arg\max}\ \boldsymbol{w}^\intercal \boldsymbol{\phi}(y, x). \tag{14}$$

The score function for inferring latent variables can be also simply represented as a summation of each localization score as below

$$s(\boldsymbol{y}, \boldsymbol{x}) = \sum_{t=1}^{T} s(y_t, x_t) = \sum_{t=1}^{T} \boldsymbol{w}^\intercal \boldsymbol{\phi}(y_t, x_t). \tag{15}$$

Even when localization system is built for localizing at once, time series locations can be predicted by maximizing the summation of scores on physically possible locations, as below.

$$\hat{\boldsymbol{y}} = \underset{\boldsymbol{y} \in \mathcal{Y}}{\arg\max} \sum_{t=1}^{T} \boldsymbol{w}^\intercal \boldsymbol{\phi}(y_t, x_t) \text{ s.t. } \boldsymbol{y} \text{ is physically possible.} \tag{16}$$

4. Empirical Evaluation

This section shows that the proposed approach is comparable to prior supervised and superior to semi-supervised approaches in real-world data.

Table 1: Calibration datasets

Dataset	Annotation	Time	# fingerprints	# annotations
L	location	5 h	9390	313
R1	route	1 h 20 min	2794	234(=3×78)
R2	route	1 h 10 min	2847	150(=3×50)
test, validation	location	40 min	604	302

4.1. Environment and Datasets

We collected calibration data on hallways of one floor. The dimensions of the floor are about 50 meters by 40 meters. The hallways are divided into 320 grids about 1 meter on a side as shown Figure 3. Location labels $\boldsymbol{l}$ are recorded with column number l_x and row number l_y ($\boldsymbol{l} = (l_x, l_y)$). A Tablet PC ThinkPad X41 (Lenovo) is used to measure RSS fingerprints and to annotate routes or location, it takes about 1 second to scan WLAN network.

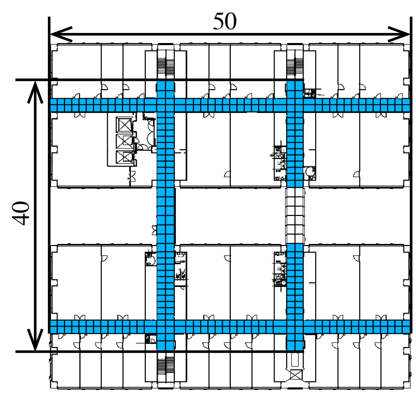

Figure 3: Environment map and grids

We collected three training datasets; one fully annotated dataset and two routes annotated datasets. On fully annotated dataset L, we successively scanned 30 times on 313 labels. Several locations were not accessible due to obstacles. Scanning on whole area took about 5 hours. We collected route-annotated data with two different number of annotation (R1 and R2). On each data, we walked around 3 laps of hallways. To compare with semi-supervised learning from partially annotated data, we also treated route data as partially annotated data that the first and the last fingerprints are labeled as the start and end points of route respectively. For validation and testing, we also collected fully annotated datasets on 302 labels. The details of data sets are summarized in Table 1.

4.2. Configurations of Proposed Methods and Comparative Methods

This section describes the implementation details of the proposed approach and comparative methods.

On the proposed approaches, we considered two cost functions Δ: binary function Δ_0 (simple Max Margin) and Manhattan distance Δ_1, as below:

$$\Delta_0(\boldsymbol{l}, \hat{\boldsymbol{l}}) = \begin{cases} 1 & \boldsymbol{l} \neq \hat{\boldsymbol{l}} \\ 0 & \boldsymbol{l} = \hat{\boldsymbol{l}} \end{cases} \tag{17}$$

$$\Delta_1(\boldsymbol{l}, \hat{\boldsymbol{l}}) = |l_x - \hat{l}_x| + |l_y - \hat{l}_y| \tag{18}$$

Another important factor is feature functions. WLAN properties can be considered as below. RSS values are obtained from beacons transmitted every 100ms by each access points. If there are many access points and their beacons do not reach to each other, they may interfere with each other. This effects whether an access point is detected or not. However, the RSS values are independent of each other. Several researches (Schwaighofer et al., 2003; Kaemarungsi and Krishnamurthy, 2004) report the probability distribution of RSS value can be represented with Gaussian distribution. Feature functions are summarized as follows:

$$\phi_d^m = \begin{cases} 1 & \text{if } m \text{ th AP detected} \\ 0 & \text{other wise} \end{cases} \quad (m = 1, \ldots, M), \tag{19}$$

$$\phi_g^m = \exp\left(-\frac{(x_m - \mu_m)^2}{\sigma_m^2}\right) \ (m = 1, \ldots, M), \tag{20}$$

$$\phi_b = 1, \tag{21}$$

where M denotes number of access points (AP), ϕ_d^m represents whether m th access point is detected or not, ϕ_g^m is a Gauss kernel feature of m th access point, x_m is RSS value of m th access point, μ_m is a mean, σ_m is a variance, ϕ_b is a dummy feature for bias. We prepared several Gauss features with different means μ and variances σ^2 for each access points. To show the appropriateness of the learning with minimizing cost-sensitive hinge-loss, we employed the update rule to fully annotated data. However, above method simply applied for fully annotated data does not show good performance. Main possible reason is that fingerprints successively measured are too nearest to learn one grid. Therefore we preprocess fully annotated data. The preprocessing is to average successive fingerprints having same labels then to add 1 grid noise to location labels.

As supervised approach for comparative methods, we implemented localization methods with k-NN, GPPS and one-versus-rest SVM. On k-NN methods, we follow NNSS-AVG algorithm proposed by Bahl et al. (2000). The algorithm picks k of nearest fingerprints and averages their locations to obtain an estimate location. This can be viewed as regression problem. The distance function for picking nearest fingerprints is Euclid distance. When access point is not detected, the RSS value is set -85dBm which is minimum value on the device. We varied k values (1, 50, 100, 150, 200, 250) then chose typical result. On Gaussian process methods, we referred (Schwaighofer et al., 2003; Ferris et al., 2006). This method learns, for each access points, probabilistic model that describes the distribution of received signal strength. The probabilistic model is obtained by Gaussian process. Then estimated position is computed by maximizing the joint likelihood. For Gaussian process, parameters of covaricance function are determined with grid search and common for all access points. The learning has cubic complexity of number of samples for each access points. Therefore, 700 samples are randomly selected. If parameters are determined by maximizing likelihood of train samples, this method may predict more accurately. However, the learning of parameter takes much computation time. We, therefore, chose grid search. On one-versus-rest SVM, treated as classification. Feature function is same with the proposed method. While we applied for both of raw and preprocessed data, the results of the raw data were slightly better. To compare with semi-supervised methods, we also implemented label propagation methods by referring to (Kashima et al., 2007; Yang et al., 2008). The computation time is cubic of number of unlabeled samples. We, so that, used grid search on fully annotated data to determine parameters of similarity. To show stability of the learning from route, we also implemented another method that basic idea is same with the proposed approach but latent-labels are inferred from partially annotated data, as mentioned Section 3.1.

4.3. Results

In this experiment, grid is used as unit of error for evaluation. The error $e(\boldsymbol{l}, \hat{\boldsymbol{l}})$ is calculated with true label $\boldsymbol{l}$ and estimated label $\hat{\boldsymbol{l}}$ as below:

$$e(\boldsymbol{l}, \hat{\boldsymbol{l}}) = \sqrt{(l_x - \hat{l}_x)^2 + (l_y - \hat{l}_y)^2}. \tag{22}$$

Table 2 shows mean errors and the 90th percentiles of error distance for each methods on each datasets.

On fully annotated data, GPPS method outperforms other methods. For example, mean error distance and 90th percentile of error distance for Max-Margin are 2.89 grids and 6.00 grids, respectively. k-NN got best performance one when $k = 150$. On L data, to estimate one grid, learning from near grids is necessary.

Max-Margin with Δ_0 and one-versus-rest SVM show worse accuracy than even k-NN($k = 1$). GPPS and k-NN use several grids to estimate one location. These results show that Max-margin methods considering location cost is appropriate for WLAN localization.

On R1, the proposed method outperforms other methods even GPPS. Moreover, the result of the proposed method on R2 is comparable to GPPS. This demonstrates route-annotated data have more useful information than fully annotated data on this situation. One of possible reasons is that near grids are properly used for learning one grid. Another reason is route annotated data can be collected on 3 laps, whereas fully annotated data has only 1 lap data. While the fully annotated data may not have sufficient amount of data, building calibration data with route is incredibly easy and quick. In addition, comparing with other semi-supervised method, the proposed method can learn from even R2 that is more sparsely annotated data.

Figure 4 plots the results of typical methods with the cumulative distribution functions (CDF) of the distance error. The result of GPPS has a long tail until over 35 grids. On the other hand, the proposed method keeps good accuracy even on R2 dataset. This shows the proposed approach can predict not only with small mean error, but also with robustness.

5. Conclusion

We proposed "route-annotation" methodology to reduce the efforts and time required to build calibration data. Also, we developed a novel learning scheme that can effectively

Table 2: Comparison with mean error and 90th percentile of error distance. The best score in each class is shown in bold.

train data	method	error distance [grids] mean	error distance [grids] 90th %tile
L	k-NN ($k = 1$)	3.91	7.07
	k-NN ($k = 150$)	3.59	7.07
	k-NN ($k = 200$)	3.62	7.28
	GPPS	**2.89**	**6.00**
	one-versus-rest SVM	5.27	13.00
	Label Propagation	3.73	7.28
	Max-Margin with Δ_0	4.09	9.06
	Max-Margin with Δ_1	3.34	6.32
R1	Label Propagation	3.03	7.00
	Max-Margin from partially annotated data with Δ_1	3.67	8.06
	Proposed with Δ_1	**2.53**	**5.10**
R2	Label Propagation	5.02	10.44
	Max-Margin from partially annotated data with Δ_1	4.06	9.00
	Proposed with Δ_1	**2.99**	**6.00**

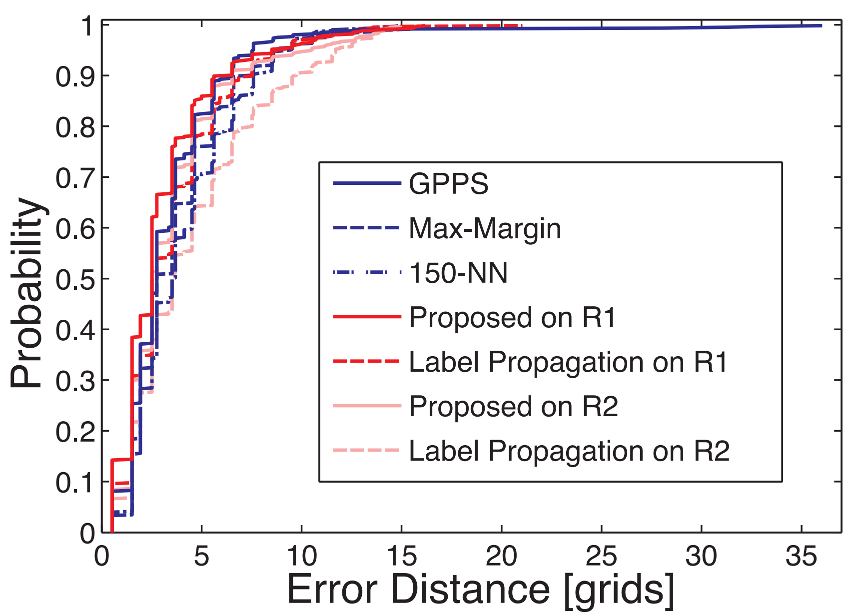

Figure 4: CDF of distance error

learn from route-annotated data. The experimental result shows the proposed method has comparative accuracy to prior approaches with much amount of data.

The problem of the current learning algorithm from route-annotated data is not assured convexity. Therefore, it is important to analyze the convexity of the problem and apply relaxation to avoid local optima. This analysis may derive proper parameter initialization and reduce learning time. Furthermore, we will tackle to deploy a building scale system in order to show the feasibility of the learning.

References

P. Bahl and V.N. Padmanabhan. RADAR: An in-building RF-based user location and tracking system. In *Proc. of INFOCOM*, 2000.

P. Bahl, V.N. Padmanabhan, and A. Balachandran. A software system for locating mobile users: Design, evaluation, and lessons. *online document, Microsoft Research*, 2000.

J. Duchi and Y. Singer. Efficient online and batch learning using forward backward splitting. *The Journal of Machine Learning Research*, 2009.

Inc. Ekahau. RTLS. URL http://www.ekahau.com/.

Pedro F. Felzenszwalb, David A. McAllester, and Deva Ramanan. A discriminatively trained, multiscale, deformable part model. In *Proc. of CVPR*, 2008.

E. Fernandes and U. Brefeld. Learning from partially annotated sequences. *Machine Learning and Knowledge Discovery in Databases*, 2011.

B. Ferris, D. Hähnel, and D. Fox. Gaussian processes for signal strength-based location estimation. In *Proc. of RSS*, 2006.

K. Kaemarungsi and P. Krishnamurthy. Properties of indoor received signal strength for WLAN location fingerprinting. In *Proc. of MOBIQUITOUS*, 2004.

H. Kashima, S. Suzuki, S. Hido, Y. Tsuboi, T. Takahashi, T. Ide, R. Takahashi, and A. Tajima. A semi-supervised approach to indoor location estimation. 2007.

B. Maeireizo, D. Litman, and R. Hwa. Co-training for predicting emotions with spoken dialogue data. In *Proc. of ACL*, 2004.

S.J. Pan, V.W. Zheng, Q. Yang, and D.H. Hu. Transfer learning for WiFi-based indoor localization. In *Proc. of AAAI Workshop*, 2008.

T. Pulkkinen, T. Roos, and P. Myllymäki. Semi-supervised learning for WLAN positioning. *Artificial Neural Networks and Machine Learning*, 2011.

C. Rosenberg, M. Hebert, and H. Schneiderman. Semi-supervised self-training of object detection models. *Seventh IEEE Workshop on Applications of Computer Vision*, 2005.

A. Schwaighofer, M. Grigoras, V. Tresp, and C. Hoffmann. GPPS: A gaussian process positioning system for cellular networks. *Advances in Neural Information Processing Systems*, 2003.

P. Sen and L. Getoor. Cost-sensitive learning with conditional markov networks. In *Proc. of ICML*, 2006.

Inc. Skyhook. Skyhook. URL http://www.skyhookwireless.com/.

I. Tsochantaridis, T. Joachims, T. Hofmann, and Y. Altun. Large margin methods for structured and interdependent output variables. *Journal of Machine Learning Research*, 2006.

Q. Yang, S.J. Pan, and V.W. Zheng. Estimating location using wi-fi. *IEEE Intelligent Systems*, 2008.

V.W. Zheng, S.J. Pan, Q. Yang, and J.J. Pan. Transferring multi-device localization models using latent multi-task learning. In *Proc. of AAAI*, 2008a.

V.W. Zheng, E.W. Xiang, Q. Yang, and D. Shen. Transferring localization models over time. In *Proc. of AAAI*, 2008b.