CrossMark

Feature Adaptive Online Sequential Extreme Learning Machine for lifelong indoor localization

Xinlong Jiang · Junfa Liu · Yiqiang Chen ·
Dingjun Liu · Yang Gu · Zhenyu Chen

Abstract Wi-Fi-based indoor localization with high capability and feasibility needs to implement lifelong online learning mechanism. However, the characteristic of Wi-Fi is wide variability, which lies in not only the fluctuation of signal strength value, but also the increase or decrease in the number of access points (APs). The traditional algorithms are effective for signal fluctuation, but cannot handle the dimension-changing problem of features caused by increase and decrease in APs' number. To solve this problem, we propose a Feature Adaptive Online Sequential Extreme Learning Machine (FA-OSELM) algorithm. It can transfer the original model to a new one with a small number of data with new features, so as to make the new model suitable for the new feature dimension. The experiments show that the FA-OSELM can get higher accuracy with a small amount of new data, and it is an effective method to make lifelong indoor localization practical.

X. Jiang (✉) · J. Liu · Y. Chen · Y. Gu · Z. Chen
Beijing Key Laboratory of Mobile Computing and Pervasive Device, Institute of Computing Technology, Chinese Academy of Sciences, Beijing 100190, China
e-mail: jiangxinlong@ict.ac.cn

X. Jiang · Y. Gu · Z. Chen
University of Chinese Academy of Sciences,
Beijing 100190, China

D. Liu
Xiangtan University, Xiangtan 411105, Hunan, China

1 Introduction

As indoor location-based service (indoor LBS) [1] has been more and more important in our daily life, indoor location estimation is becoming the key problem. At present, the most popular techniques implementing indoor location is Wi-Fi-based location fingerprinting [2–4]. It has the major advantage of exploiting existing wireless network infrastructures and consequently avoiding extra deployment costs.

Location fingerprinting requires to operate a spatial signal strength map from different access points (APs) strategically located in a given area. It becomes a classical classification problem where different supervised machine learning techniques have been used to train classifiers, using the signal strength from different APs as their input data and providing the location estimation as their output estimation. K-NN (nearest neighbor) [5], decision tree [6], Bayesian [7], neural networks [8], extreme learning machine (ELM) [9] are most frequently used by location fingerprinting. Among all the algorithms above, ELM is more and more widely used for its competitive fast learning speed during both offline and online phrases.

Nevertheless, due to Wi-Fi signal is dynamically changing over time [10], the location accuracy decreases as time goes on. The dynamism of Wi-Fi signals is diverse.

On the one hand, the dynamism means the signal strength value, which is caused by the volatility of Wi-Fi signal and the change of environment. It will lead occasional received signal strength indication (RSSI) value missing, as shown in Table 1. For this situation, the general approach is to supplement it with a default value according to the data's distribution. Chen [11] sets all the missing values to −95, the minimum strength of the signal received in the environment. Roos [12] replaces the missing values

Table 1 Wi-Fi location fingerprints

ID	RSSI				
	AP_1	AP_2	AP_3	…	AP_n
1	−85	−81	−67	…	N/A
2	−79	−83	N/A	…	−84
3	N/A	−73	−65	…	−80
…	…	…	…	…	…
m	−88	N/A	−70	…	−87

with some constant smaller than any of the measured values.

On the other hand, the dynamism also denotes the number of APs. Due to the basic function of Wi-Fi APs is to provide Internet, it is very common that some APs are removed or some new APs are added in the environment, shown in Fig. 1. This situation will be difficult to solve when the missing APs offer the fingerprint information, or the new coming APs will be used as the feature. It will bring the change of feature dimension, which is a challenge for traditional machine learning algorithms. By using these traditional methods, such as support vector machine (SVM) [4, 13] and ELM, we can do nothing but collect new data and retrain a model, it requires a lot of extra computation and labor costs. At the same time, there are no direct machine learning algorithms, which can handle the training data of varying feature dimension.

Focusing on this problem, we regard it as a feature transfer learning problem and propose a Feature Adaptive

Online Sequential Extreme Learning Machine (FA-OSELM) algorithm. It can transfer the original model to a new one with a few of incremental data, rather than completely retrain a new model. The experiments show that the transferred model can get high accuracy.

The rest of the paper is organized as follows. We firstly review ELM and OS-ELM briefly in Sect. 2. Then in Sect. 3, we introduce FA-OSELM in detail. After that, we do some experiments and evaluate the performance in Sect. 4. At last, in Sect. 5, we make a short conclusion.

2 Brief of ELM and OS-ELM

In this section, we review ELM [9, 14, 15] and OS-ELM [16] algorithms by introducing their motivation, modeling and algorithm steps.

ELM is developed by Huang et al. And it belongs to artificial neural network (ANN) family, especially an single-layer feedforward networks (SLFN), where learning is made without iterative tuning. According to ELM learning theory [17], if SLFNs $f(\mathbf{x}) = h(\mathbf{x})\beta$ with tunable *piecewise* continuous hidden-layer feature mapping $h(\mathbf{x})$ can approximate any target continuous functions, tuning is not required in the hidden layer then. All the hidden-node parameters, which are supposed to be tuned by conventional learning algorithms, can be randomly generated according to any continuous sampling distribution [18, 19].

Comparing with other traditional learning methods, ELM has not only better performance in classification

Fig. 1 The increase or decrease in the APs number. **a** AP_6 is removed from the environment, which will lead decreasing of the feature dimension, **b** a new AP_7 is added in the area, which can offer positive influence on location accuracy

(a)

(b)
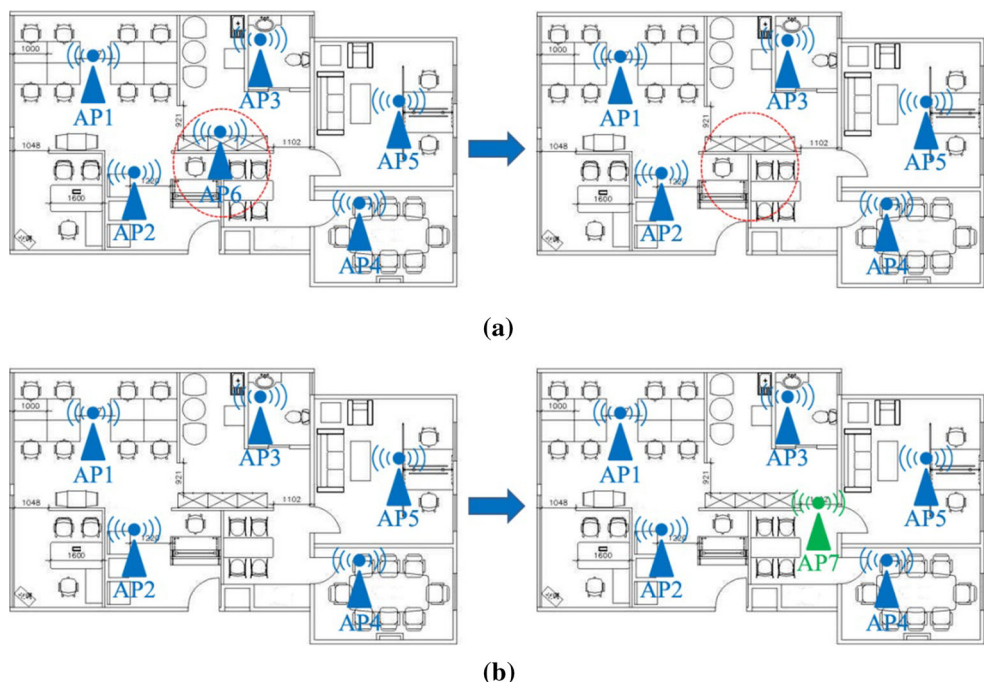
precision and regression fitting degree, but also less time consumption in offline learning and online prediction [20].

Now, more and more work has been done to develop ELM. Andrés [21] and Wang [22] propose some new method to improve the generalization capability of ELM. Miche [23] improves the optimally pruned extreme learning machine (OP-ELM) with LARS and Tikhonov regularization to a double-regularized ELM. Therefore, it can maintain numerical stability and efficient pruning of the neurons. Dealing with some problems, especially relatively large datasets, ELM suffers from instability and over-fitting. Zhai et al. [24] propose an approach of fusion of extreme learning machine (F-ELM) with fuzzy integral based on probabilistic SLFNs.

And in order to alleviate some extent the problems of instability and over-fitting problems of ELM when dealing with large datasets, Zhai et al. [25] propose a dynamic ensemble extreme learning machine based on sample entropy. The experimental results show that the proposed approach is robust and efficient. Furthermore, ELM has been widely utilized in various kinds of applications such as indoor localization [26, 27], activity recognition [28, 29], transportation mode recognition [30, 31], context-aware computing [32] and so on.

Given N arbitrary distinct samples $(\mathbf{x}_i, \mathbf{t}_i) \in R^n \times R^m$, $i = 1, 2, \ldots, N$. Here, $\mathbf{x}_i$ is a $n \times 1$ input vector $\mathbf{x}_i = [x_{i1}, x_{i2}, \ldots, x_{in}]^T$ and $\mathbf{t}_i$ is a $m \times 1$ target vector $\mathbf{t}_i = [t_{i1}, t_{i2}, \ldots, t_{im}]^T$. The network with L hidden nodes is shown in Fig. 2. The output function of this network can be represented as follows:

$$f_L(\mathbf{x}_j) = \sum_{i=1}^{L} \beta_i G(\mathbf{a}_i, b_i, \mathbf{x}_j), \quad j = 1, \ldots, N \tag{1}$$

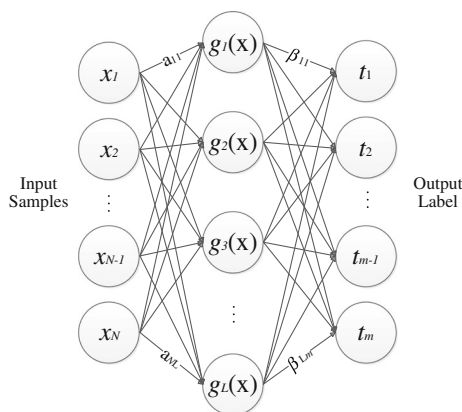where $\mathbf{a}_i$ and b_i are the learning parameters of hidden nodes, and β_i is the weight connecting the ith hidden node

to the output node. $G(\mathbf{a}_i, b_i, \mathbf{x})$ is the output of the ith hidden node with respect to the input x. For additive hidden node with the activation function $g(x): R \rightarrow R$ (e.g., sigmoid and threshold), $G(\mathbf{a}_i, b_i, \mathbf{x})$ is given by

$$G(\mathbf{a}_i, b_i, \mathbf{x}) = g(\mathbf{a}_i \cdot \mathbf{x} + b_i), \quad b_i \in R \tag{2}$$

If an SLFN with L hidden nodes can approximate these N samples with zero error, it then implies that there exist $\beta_i, \mathbf{a}_i$ and b_i such that

$$f_L(\mathbf{x}_j) = \sum_{i=1}^{L} \beta_i G(\mathbf{a}_i, b_i, \mathbf{x}_j) = \mathbf{t}_j, \quad j = 1, \ldots, N. \tag{3}$$

Equation (3) can be summarized as

$$H\beta = T \tag{4}$$

where

$$H(\mathbf{a}_1, \ldots, \mathbf{a}_L, b_1, \ldots, b_L, \mathbf{x}_1, \ldots, \mathbf{x}_L)$$
$$= \begin{bmatrix} G(\mathbf{a}_1, b_1, \mathbf{x}_1) & \cdots & G(\mathbf{a}_L, b_L, \mathbf{x}_1) \\ \vdots & \ddots & \vdots \\ G(\mathbf{a}_1, b_1, \mathbf{x}_N) & \cdots & G(\mathbf{a}_L, b_L, \mathbf{x}_N) \end{bmatrix} \tag{5}$$

$$\beta = \begin{bmatrix} \beta_1^T \\ \vdots \\ \beta_L^T \end{bmatrix}_{L \times m} \quad \text{and} \quad T = \begin{bmatrix} \mathbf{t}_1^T \\ \vdots \\ \mathbf{t}_N^T \end{bmatrix}_{N \times m} \tag{6}$$

According to [10], the hidden-node parameters $\mathbf{a}_i$ and b_i (input weights and biases or centers and impact factors) of SLFNs do not need to be tuned during training and may simply be assigned with random values. The smallest norm least-squares solution of the above linear system is

$$\hat{\beta} = H^{\dagger}T \tag{7}$$

where $H^{\dagger}$ is the Moore–Penrose generalized inverse of matrix H [33, 34]. Different methods can be used to calculate the Moore–Penrose generalized inverse of a matrix: orthogonal projection method, orthogonalization method, iterative method and singular value decomposition (SVD) [34]. The orthogonal projection method [34] can be used in two cases: when $H^T H$ is nonsingular and $H^{\dagger} = (H^T H)^{-1} H^T$ or when HH^T is nonsingular and $H^{\dagger} = H^T (HH^T)^{-1}$.

2.1 OS-ELM

The batch ELM described previously assumes that all the training data (N samples) are available for training. However, in real applications, the training data may arrive chunk by chunk or one by one (a special case of chunk), and hence, the batch ELM algorithm has to be modified for this case so as to make it online sequential.

First, given a chunk of initial training set $\aleph_0 = \{(\mathbf{x}_i, \mathbf{t}_i)\}_{i=1}^{N_0}$ and $N_0 \geq L$, if one considers using the batch

Fig. 2 SLFN with L hidden nodes

ELM algorithm, one need to consider only the problem of minimizing $\|H_0\beta - T_0\|$. According to [16], the solution to minimizing $\|H_0\beta - T_0\|$ is given by $\beta^{(0)} = K_0^{-1}H_0^T T_0$ where $K_0 = H_0^T H_0$.

Now support that we are given another chunk of data $\aleph_1 = \{(\mathbf{x}_i, \mathbf{t}_i)\}_{i=N_0+1}^{N_0+N_1}$, where N_1 denotes the number of observations in the chunk, the problem then becomes minimizing

$$\left\| \begin{bmatrix} H_0 \\ H_1 \end{bmatrix} \beta - \begin{bmatrix} T_0 \\ T_1 \end{bmatrix} \right\| \tag{8}$$

where

$$H_1 = \begin{bmatrix} G(\mathbf{a}_1, b_1, \mathbf{x}_{N_0+1}) & \cdots & G(\mathbf{a}_L, b_L, \mathbf{x}_{N_0+1}) \\ \vdots & \ddots & \vdots \\ G(\mathbf{a}_1, b_1, \mathbf{x}_{N_0+N_1}) & \cdots & G(\mathbf{a}_L, b_L, \mathbf{x}_{N_0+N_1}) \end{bmatrix}_{N_1 \times L},$$

$$T_1 = \begin{bmatrix} \mathbf{t}_{N_0+1}^T \\ \vdots \\ \mathbf{t}_{N_0+N_1}^T \end{bmatrix}_{N_1 \times m} \tag{9}$$

Considering both chunks of training data sets $\aleph_0$ and $\aleph_1$, the output weight β is formulated as

$$\beta^{(1)} = K_1^{-1} \begin{bmatrix} H_0 \\ H_1 \end{bmatrix}^T \begin{bmatrix} T_0 \\ T_1 \end{bmatrix} \tag{10}$$

where

$$K_1 = \begin{bmatrix} H_0 \\ H_1 \end{bmatrix}^T \begin{bmatrix} H_0 \\ H_1 \end{bmatrix} \tag{11}$$

and

$$\begin{bmatrix} H_0 \\ H_1 \end{bmatrix}^T \begin{bmatrix} T_0 \\ T_1 \end{bmatrix} = H_0^T T_0 + H_1^T T_1 = K_0 K_0^{-1} H_0^T T_0$$
$$+ H_1^T T_1 = K_0 \beta^{(0)} + H_1^T T_1$$
$$= (K_1 - H_1^T H_1)\beta^{(0)} + H_1^T T_1 = K_1\beta^{(0)}$$
$$- H_1^T H_1 \beta^{(0)} + H_1^T T_1 \tag{12}$$

So the model of OS-ELM will be updated by the incremental data. The contribution of incremental data x^* is reflected by the correction $\Delta\beta$, of existing parameter of training model β', which forms the new parameter of training model β^* as Eq. (13).

$$\beta^* = \beta' + \Delta\beta(x^*) \tag{13}$$

Obviously, the result of β^* is based on previous result β', but the calculation burden is not heavy for it do not need all the data to retrain the model.

Combining (10) and (12), $\beta^{(1)}$ is given by

$$\beta^{(1)} = K_1^{-1} \begin{bmatrix} H_0 \\ H_1 \end{bmatrix}^T \begin{bmatrix} T_0 \\ T_1 \end{bmatrix}$$
$$= K_1^{-1}\left(K_1\beta^{(0)} - H_1^T H_1\beta^{(0)} + H_1^T T_1\right)$$
$$= \beta^{(0)} + K_1^{-1}H_1^T\left(T_1 - H_1\beta^{(0)}\right) \tag{14}$$

where

$$K_1 = \begin{bmatrix} H_0 \\ H_1 \end{bmatrix}^T \begin{bmatrix} H_0 \\ H_1 \end{bmatrix} = \begin{bmatrix} H_0^T & H_1^T \end{bmatrix} \begin{bmatrix} H_0 \\ H_1 \end{bmatrix} = K_0 + H_1^T H_1 \tag{15}$$

Obviously, Eq. (14) corresponds to Eq. (13), the new $\beta^{(1)}$ is drives from $\beta^{(0)}$, we only need the new coming incremental data to update the $\beta^{(0)}$ to $\beta^{(1)}$. So we greatly reduce the computation cost because only a few data are used for updating.

With the incremental number increasing, when the $(k+1)$th chunk of data set $\aleph_{k+1} = \{(\mathbf{x}_i, \mathbf{t}_i)\}_{i=\left(\sum_{j=0}^{k} N_j\right)+1}^{\sum_{j=0}^{k+1} N_j}$ is received, where $k \geq 0$ and N_{k+1} denotes the number of observations in the $(k+1)$th chunk, the Eq. (14) for updating $\beta^{(k+1)}$ will be written as

$$\beta^{(k+1)} = \beta^{(k)} + K_{k+1}^{-1}H_{k+1}^T\left(T_{k+1} - H_{k+1}\beta^{(k)}\right) \tag{16}$$

$K_{k+1}^{-1}H_{k+1}^T\left(T_{k+1} - H_{k+1}\beta^{(k)}\right)$ can be seen as the correction of the original model $\beta^{(k)}$ with the new samples $\aleph_{k+1} = \{(\mathbf{x}_i, \mathbf{t}_i)\}_{i=\left(\sum_{j=0}^{k} N_j\right)+1}^{\sum_{j=0}^{k+1} N_j}$.

3 FA-OSELM

In Sect. 2, we have reviewed the ELM and OS-ELM algorithms. Fortunately, they have already been applied indoor localization research area. Xiao et al. [35] achieved a perfect performance in Wi-Fi indoor location using ELM. But due to the basic function of Wi-Fi is to provide Internet, it is very common that some APs are removed or some new APs are added. This change will affect the variation of feature dimension, and thus, the old model will not work anymore.

Considering the problem, we propose a FA-OSELM. When the number of features is changed, we can update the model using a small amount of incremental data with new features.

Firstly, given N_0 arbitrary distinct samples $(\mathbf{x}_i, \mathbf{t}_i) \in R^n \times R^m, i = 1, 2, \ldots, N_0$. Here, $\mathbf{x}_i$ is a $n \times 1$ input vector $\mathbf{x}_i = [x_{i1}, x_{i2}, \ldots, x_{in}]^T$ and $\mathbf{t}_i$ is a $m \times 1$ target vector $\mathbf{t}_i = [t_{i1}, t_{i2}, \cdots, t_{im}]^T$.

When the number of APs used as the features changes, we can collect a new batch data to update the model. Given another N_1 arbitrary distinct samples $\left(\mathbf{x}_i', \mathbf{t}_i\right) \in R^{n'} \times R^m, i = 1, 2, \ldots, N_1$. Here, $\mathbf{x}_i'$ is a $n' \times 1$ input vector $\mathbf{x}_i' = \left[x_{i1}', x_{i2}', \ldots, x_{in'}'\right]^T$, but $\mathbf{t}_i$ is still a $m \times 1$ target vector $\mathbf{t}_i = [t_{i1}, t_{i2}, \ldots, t_{im}]^T$.

If $n' < n$, it means some APs in the location area are missing; if $n' > n$, it means that we set some new APs to the location area. In these two situations, the problem is still to minimize

$$\left\| \begin{bmatrix} H_0 \\ H_1 \end{bmatrix} \beta - \begin{bmatrix} T_0 \\ T_1 \end{bmatrix} \right\| \tag{17}$$

But the H_0, H_1, T_0 and T_1 refer the follows:

$$H_0 = \begin{bmatrix} G(\mathbf{a}_1, b_1, \mathbf{x}_1) & \cdots & G(\mathbf{a}_L, b_L, \mathbf{x}_1) \\ \vdots & \ddots & \vdots \\ G(\mathbf{a}_1, b_1, \mathbf{x}_{N_0}) & \cdots & G(\mathbf{a}_L, b_L, \mathbf{x}_{N_0}) \end{bmatrix}_{N_0 \times L},$$

$$T_0 = \begin{bmatrix} \mathbf{t}_1^T \\ \vdots \\ \mathbf{t}_{N_0}^T \end{bmatrix}_{N_0 \times m} \tag{18}$$

$$H_1 = \begin{bmatrix} G(\mathbf{a}_1', b_1, \mathbf{x}_{N_0+1}') & \cdots & G(\mathbf{a}_L', b_L, \mathbf{x}_{N_0+1}') \\ \vdots & \ddots & \vdots \\ G(\mathbf{a}_1', b_1, \mathbf{x}_{N_0+N_1}') & \cdots & G(\mathbf{a}_L', b_L, \mathbf{x}_{N_0+N_1}') \end{bmatrix}_{N_1 \times L},$$

$$T_1 = \begin{bmatrix} \mathbf{t}_{N_0+1}^T \\ \vdots \\ \mathbf{t}_{N_0+N_1}^T \end{bmatrix}_{N_1 \times m} \tag{19}$$

where

$$\left\{\mathbf{a}_i = \{a_1, a_2, \ldots, a_n\}\right\}_{i=1}^L, \quad \left\{\mathbf{x}_i = \{x_1, x_2, \ldots, x_n\}\right\}_{i=1}^{N_0} \tag{20}$$

$$\left\{\mathbf{a}_i' = \{a_1', a_2', \ldots, a_{n'}'\}\right\}_{i=1}^L, \quad \left\{\mathbf{x}_i' = \{x_1', x_2', \ldots, x_{n'}'\}\right\}_{i=1}^{N_1} \tag{21}$$

$\mathbf{a}_i$ is the weight vector connecting the input layer to the ith hidden node, and b_i is the bias of the ith hidden node. $\mathbf{a}_i \cdot \mathbf{x}_i$ denotes the inner product of vectors $\mathbf{a}_i$ and $\mathbf{x}_i$ in R^n. $\mathbf{a}_i$ and b_i can be random generated. Once determined, they cannot be changed any more. According to Eqs. (18) and (19), we can see that $\mathbf{a}_i$ has the same dimension with $\mathbf{x}_i$, $\mathbf{a}_i'$ has the same dimension with $\mathbf{x}_i'$. Actually, $\mathbf{a}_i$ and $\mathbf{x}_i$ have one by one corresponding relationship for each column, so as to $\mathbf{a}_i'$ and $\mathbf{x}_i'$.

As shown in Fig. 3, when the feature dimension is changed, the bone structure of network has no change. But as the feature dimension is different from previous, we have to adjust $\mathbf{a}_i$ to fit new feature dimension. As the same
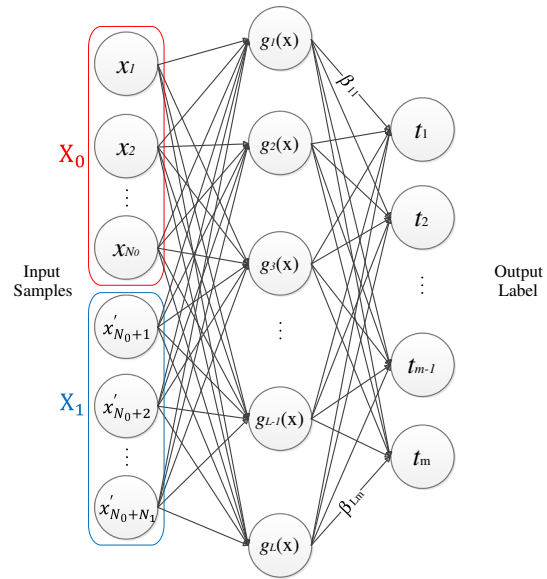

Fig. 3 FA-OSELM network

time, the hidden node has no change, so the b_i will not change.

Therefore, we propose a input-weight transfer matrix P, and a input-weight supplement vector Q_i to generate a_i' by Eq. (22).

$$\left\{\mathbf{a}_i' = \mathbf{a}_i \cdot P + \mathbf{Q}_i\right\}_{i=1}^L \tag{22}$$

where

$$P = \begin{bmatrix} P_{11} & \cdots & P_{1n'} \\ \vdots & \ddots & \vdots \\ P_{n1} & \cdots & P_{nn'} \end{bmatrix}_{n \times n'} \tag{23}$$

$$\mathbf{Q}_i = \begin{bmatrix} Q_1 & \cdots & Q_{n'} \end{bmatrix}_{1 \times n'} \tag{24}$$

matrix P has the following rules:

- Each line has only one '1,' and the rest are all '0';
- Each column has one '1' at most, and the rest are all '0';
- If $P_{ij} = 1$, it means that after the change of feature dimension, the ith dimension of the original feature vector has become the jth dimension of the new feature vector.

$\mathbf{Q}_i$ is used to supplement when the feature dimension increases, we need to add the corresponding InputWeight for the new adding features. $\mathbf{Q}_i$ has the following rules:

- When feature dimension decreased, the $\mathbf{Q}_i$ is an all-zero vector, that is to say, we do not need to add corresponding InputWeight for the new adding features;
- When feature dimension increased, if the ith item of $\mathbf{a}_i'$ is new feature, the ith item of $\mathbf{Q}_i$ should be

generated randomly according to the distributing of $\mathbf{a}_i$.

We take the Wi-Fi APs as an example:

if features: $\{Ap_1, Ap_2, Ap_3, Ap_4, Ap_5\} \rightarrow \{Ap_1, Ap_2, Ap_3, Ap_5\}$:

we can generate:

$$P = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, Q_i = \{0, 0, 0, 0\};$$

if features: $\{Ap_1, Ap_2, Ap_3, Ap_4, Ap_5\} \rightarrow \{Ap_1, Ap_2, Ap_3, Ap_6, Ap_4, Ap_5\}$:

we can generate:

$$P = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}, Q_i = \{0, 0, 0, Q_4, 0, 0\},$$

where Q_4 can be generated randomly.

As mentioned above, FA-OSELM can be summarized in the following steps:

1. Determine the model parameters by the original dataset of N_0 samples, such as the number of hidden nodes L and the activation function $g(x)$.
2. Randomly assign the value of weight vector $\mathbf{a}_i$ and bias scalar b_i, $i = 1, 2, \ldots, L$.
3. Calculate the original hidden-layer output matrix H_0.
4. Calculate the initial model parameter $\beta^{(0)} = H_0^\dagger T_0$.
5. When coming N_1 samples data X_1, T_1 with different feature, generate the input-weight transfer matrix P and input-weight supplement vector $\mathbf{Q}_i$, $i = 1, 2, \ldots, L$ according to rules mentioned above.
6. Calculate the new weight vector $\mathbf{a}_i' = \mathbf{a}_i \cdot P + \mathbf{Q}_i$, $i = 1, 2, \ldots, L$.
7. Divide the newly incremental data into k parts, set $j = 1$, then go to iterative process.
8. Using the new weight vector $\mathbf{a}_i'$ to calculate the jth iteration of model parameter H_j by Eq. (19).
9. Calculate the β_j by the Eq. (16).
10. If $j < k$, set $j = j + 1$ and go to (8); else go to (11).
11. After k times iteration, we can get the final parameter $\beta^* = \beta_j$.

From Step (5) to Step (10), when a new batch of data comes, we will adjust the weight vector basing on the change of features. So we name our algorithm FA-OSELM. The workflow of the algorithm can be concisely given in Fig. 4.
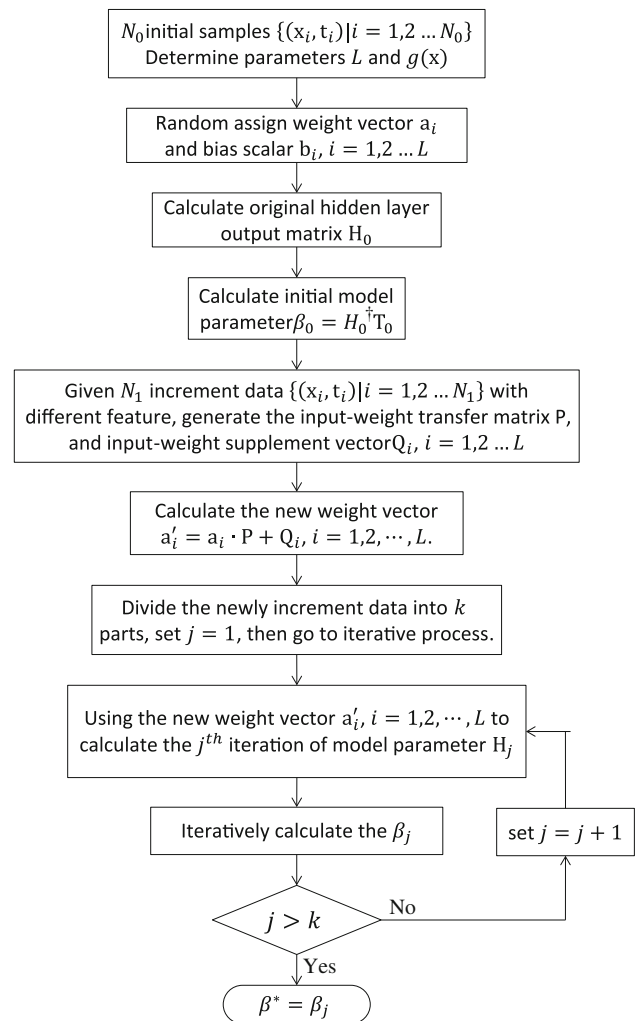
Fig. 4 The workflow of algorithm

4 Experiments and performance evaluation

As the Wi-Fi access points always move, which will leads the change of feature dimension, we use FA-OSELM to enable the existing model to overcome it with a small amount of incremental data, saving human labeling work and time-consuming.

All the experiments are running on the computer with following configuration:

Operation System	Windows XP Professional SP3
CPU	Intel Pentium(R) 4 CPU
Main Frequency	3.2 GHz
RAM	2G

4.1 Data preparing

For the classification studies, four benchmark problems have been considered: two Wi-Fi indoor location data sets:

(1) office area dataset and (2) lounge area dataset; two UCI [36] datasets: (3) image segment and (4) satellite image. We use the Wi-Fi datasets to show FA-OSELM has good performance on lifelong indoor localization. Meanwhile, the experiments on UCI datasets show that FA-OSELM is also effective in other applications.

Office area is a 12×6 m^2 working space in eighth floor of our institute, shown in Fig. 5. Red points are the locations where data are mostly collected with distance about 2 m. We collect the data at different time of a day and last for a month. Finally, we collected 5,635 data. We choose the most stable 7 APs as the feature, so each fingerprint is a seven dimension vector.

Lounge area is an 8.7×55 m^2 space in first floor of our institute, shown in Fig. 6. Also the red points show the locations where data are mostly collected with distance 2–3 m. A total of 2,484 data are collected for a week, and 18APs are selected as the feature.

The image segmentation problem consists of a database of images drawn randomly from seven outdoor images and consists of 2,310 regions of 3×3 pixels. The goal is to recognize each region into one of the seven paths, and grass using 19 attributes extracted from each square region.

The satellite image problem consists of a database generated from landsat multispectral scanner. One frame of landsat multispectral scanner imagery consists of four digital images of the same scene in four different spectral bands. The database is a (tiny) subarea of a scene, consisting of 82×100 pixels. Each data in the database corresponds to a region of 3×3 pixels. The aim is to classify of the central pixel in a region into the six categories,
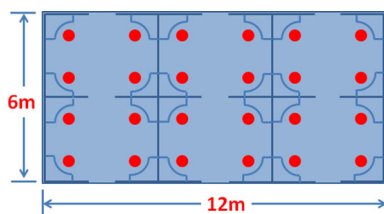
Table 2 Specifications of two UCI data sets

Dataset	# Attributes	# Classes	# Data
Image segment	19	7	2,310
Satellite image	36	6	6,435

namely red soil, cotton crop, gray soil, damp gray soil, soil with vegetation stubble, and very damp gray soil using 36 spectral values for each region.

Specifications of two UCI datasets are shown in Table 2. Before being used, they were normalized with z-score method.

4.2 Experimental performance

4.2.1 Model selection

According to Huang [20], the accuracy will be improved with regularization factor C, which can optimize the architecture of learning model. Thus, for FA-OSELM, only two user-specified parameters, regularization factor and number of hidden nodes (C, L), should be determined.

We divide the training data of each dataset into two equal subsets and use cross-validation method to determine user-specified parameters (C, L), where C is chosen from the range $\{2^{-20}, 2^{-18}, \ldots, 2^{18}, 2^{20}\}$ and L is chosen from the range $\{10, 20, \ldots, 990, 1,000\}$. The performance of each data sets' parameters is illustrated in Figs. 7, 8, 9 and 10.

From all above mentioned in Figs. 7, 8, 9 and 10, for different datasets, we need to select the optimal parameter settings of L and C to achieve good performance. For example in Fig. 7, we can obtain the optimal performance with the optimal parameter pair $(L = 350, C = 2^{-6})$ in Table 3. Generally speaking, the performance can be reached the higher accuracy increasingly as long as the number of hidden nodes L is getting larger, but due to the over-fitting problem, the accuracy will decrease in some cases in Figs. 8 and 9. Meanwhile, regularization factor
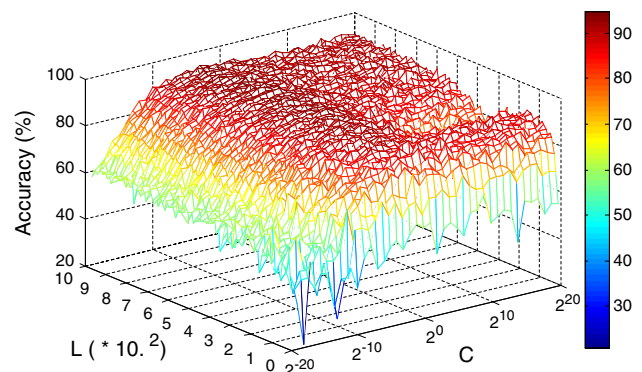

Fig. 5 Wi-Fi indoor location (office area)
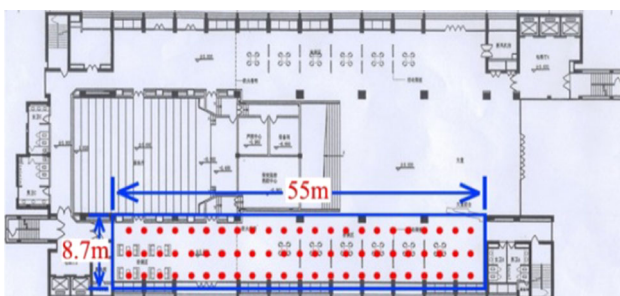
Fig. 6 Wi-Fi indoor location (lounge area)

Fig. 7 Performances with different user-specified parameters (C, L) for image segment dataset
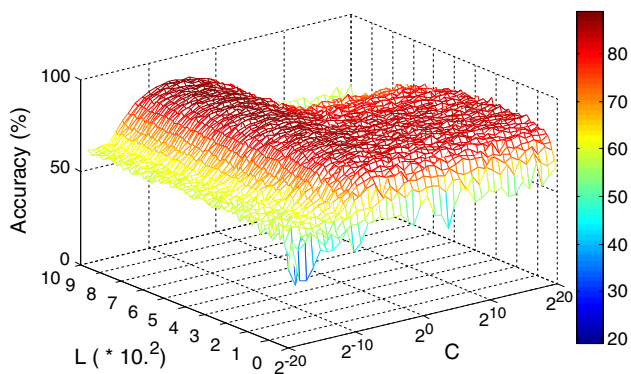
Fig. 8 Performances with different user-specified parameters (*C*, *L*) for satellite image dataset
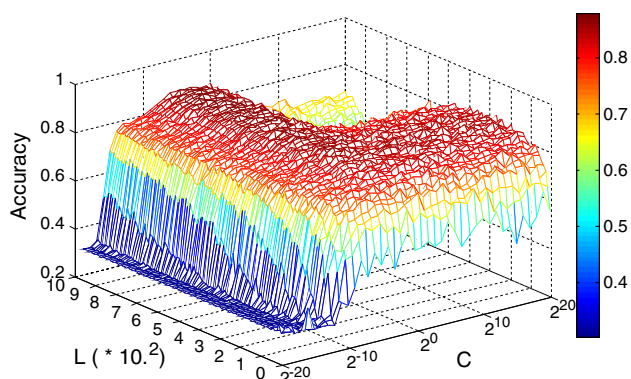

Fig. 9 Location accuracy (<5 m) with different user-specified parameters (*C*, *L*) for Wi-Fi location dataset (office area)
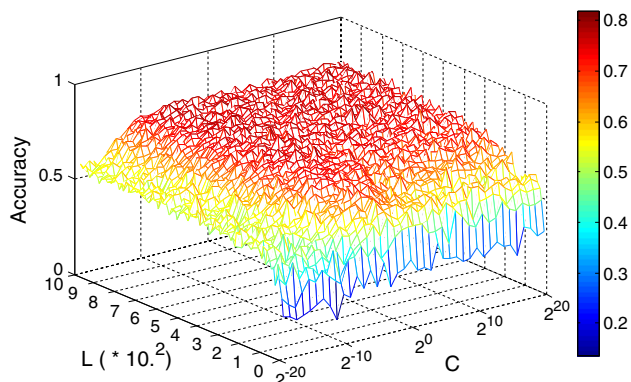

Fig. 10 Location accuracy (<15 m) with different user-specified parameters (*C*, *L*) for Wi-Fi location dataset (lounge area)

C also can help to achieve high accuracy when taking the specific value. To tradeoff between high accuracy and computation complexity, we obtain the optimal parameter settings for all the datasets in Table 3. Additionally, we use the *RBF* activation function in all experiments.

Table 3 User-specified parameters

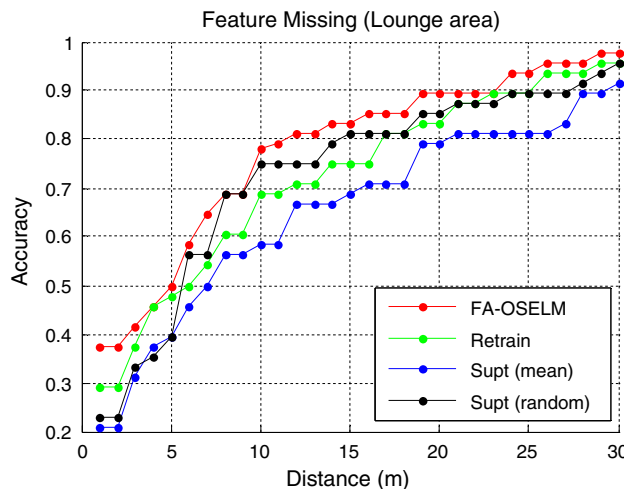| Dataset	L	C
Image segment	350	2^{-6}
Satellite image	300	2^{-10}
Wi-Fi location (office area)	400	2^{-8}
Wi-Fi location (lounge area)	400	2^{-12}

Fig. 11 Location accuracy of feature missing experiment (lounge area dataset)

4.2.2 FA-OSELM's performance when dealing with feature dimension reducing

In the study of Wi-Fi indoor localization, when one AP we used as the feature is missing, it will lead one feature missing. For this case, the old ELM model cannot be used any more and there are two common handing methods: (1) According to the distribution of the missing feature, we can fulfill the lost item with a default value, such as mean value or a random value; (2) train another new model with some new offline training data. Besides, we can use FA-OSELM to update the old model to a new one with a few of incremental data.

We apply all the methods on four datasets. We divide each dataset into three parts: training data, incremental data and testing data. Incremental data and testing data have the same feature dimension, but one less than the training data. This missing one feature dimension is random selected from the original (Figs. 11, 12).

The results of two UCI datasets are shown in Table 4. We can see that FA-OSELM has a better performance than retraining a new model. The reason is that when one feature is lost, the original model still contains the majority information of the new feature. However, the incremental data are too little to retain a good new model, but it is

Table 4 Testing accuracy in feature missing experiments

Datasets	Data proportion (training:incremental:testing)	Feature dimension			Testing accuracy			
		Training data	Incremental data	Testing data	FA-OSELM (%)	Retrain (%)	Supplement (mean value) (%)	Supplement (random value) (%)
Image segment	3:1:1	19	18	18	94.51	89.63	92.29	92.99
Satellite image	5:1:1	36	35	35	86.22	82.79	83.44	81.90

Table 5 Testing accuracy in feature increasing experiments

Datasets	Data proportion (training:incremental:testing)	Feature dimension			Testing accuracy		
		Training data	Incremental data	Testing data	FA-OSELM (%)	Retrain (%)	Old model (%)
Image segment	3:1:1	18	19	19	93.50	88.81	91.99
Satellite image	5:1:1	35	36	36	86.78	82.42	84.23

effective to transfer the original model and overcome the change of the features. FA-OSELM also performs better than the methods of supplementing with a mean value or random values because the supplemented values only enable the old model, but offer meaningless information.

4.2.3 FA-OSELM's performance when dealing with feature dimension increasing

On the contrary, when a new AP is equipped in the location area, it can offer some new feature information, but the old ELM model cannot involve it, as the feature dimension will be changed. If we want to utilize it, we can do nothing but collect a bench of data and train a new model, which needs extra labor cost. Otherwise, we have to just ignore it and still use the old model. Fortunately, FA-OSELM allows us to transfer the old model to a new one, and it can take full use of the new feature dimension with a little labor cost. As the former experiments, we also divide each datasets into three parts: training data, incremental data and testing data. But in order to test the performance when feature dimension increases, we set the feature dimension of incremental data equals to testing data, but one more than the training data.

The results of UCI datasets are shown in Table 5. We can obtain the same conclusion that FA-OSELM still works well in feature increasing situation. We can explain the low accuracy of retraining a new model as the underfitting caused by less scale of incremental data. But for FA-OSELM, the little scale of incremental data can further bring in the incremental information to the old model and improve the testing accuracy. Comparing with using the old model directly, FA-OSELM offers limited accuracy increasing for the two UCI datasets. The reason is that it is

variable that how much influence each feature can affect the accuracy.

As shown in Figs. 13 and 14, FA-OSELM's performances are much better than the other two for the Wi-Fi location problems. The reason is that, for Wi-Fi location problem, FA-OSELM can not only maintain the old model's information, but also take full of the new added feature.

4.2.4 FA-OSELM's performance as more and more incremental data comes

While adapt to the new feature dimension, FA-OSELM performs better than any other methods mentioned above with a small amount of incremental data. For the Wi-Fi-based indoor localization problem, if we can get more incremental data, we will have more information about the new location environment. We want to value whether FA-OSELM is stable and can get better performance with more incremental data.

Thus, we extend the Wi-Fi location dataset of office area to be 6,835 with another 1,200 data. We maintain the original training data and select 600 of the new data to be the testing data, and the rest are used as incremental data. They are ordered chronologically and divided into ten equal parts. We design the experiments as the previous two to measure the capability in two situations: when a feature is missing and a new feature is added. The results are shown in Figs. 15 and 16.

As shown in Figs. 15 and 16, as more incremental data chunks come, the accuracy will keep stable after a period of increasing. Because more incremental data offer more positive information to help FA-OSELM to transfer the old model so as to fit the new feature. When
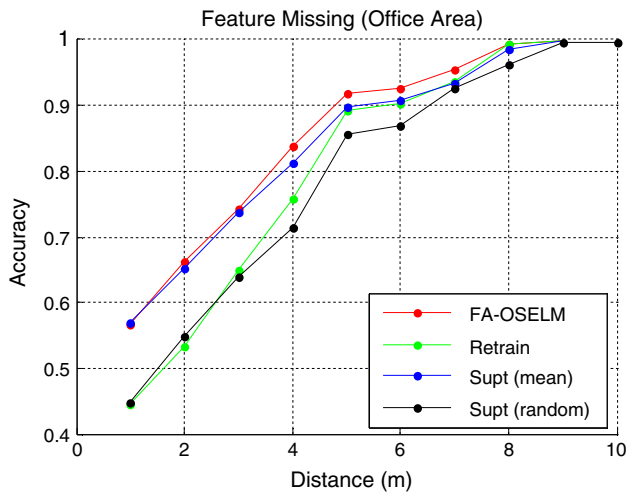
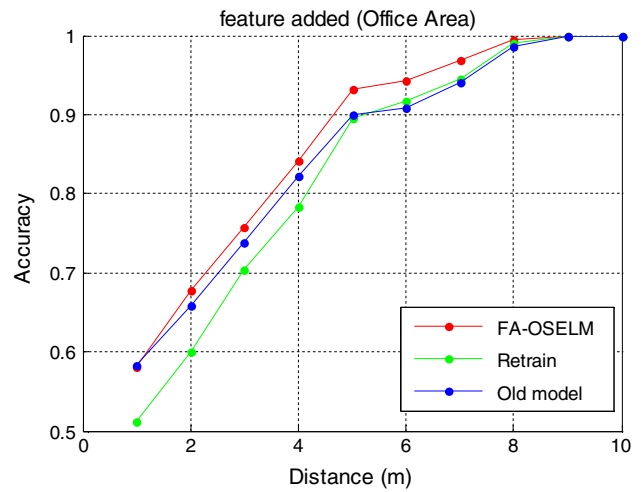**Fig. 12** Location accuracy of feature missing experiment (office area dataset)

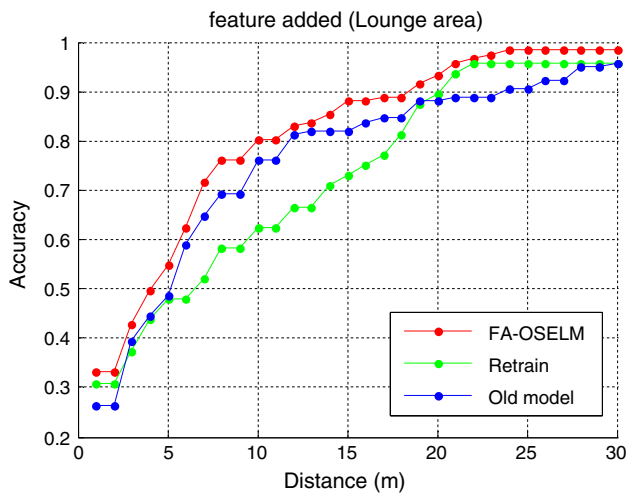**Fig. 14** Location accuracy of feature increasing experiment (office area dataset)

Fig. 13 Location accuracy of feature increasing experiment (lounge area dataset)

Fig. 15 Accuracy increasing as incremental data comes chunk by chunk in feature missing case (office area dataset)

the accuracy keeps stable, it means the new model has already overcome the accuracy gap caused by the change of the feature.

5 Conclusion

In this paper, we proposed a FA-OSELM method for lifelong indoor localization. Why we raise this problem is that: Indoor location estimate based on Wi-Fi is difficult for the sake of the APs' high dynamics. The high dynamics means not only the APs' signal strength, but also the increase or decrease in APs' number. If the lack of signal strength is accidental, we can supplement with a default value. But if the APs used as the features are missing (maybe someone removed them), or we set new APs in
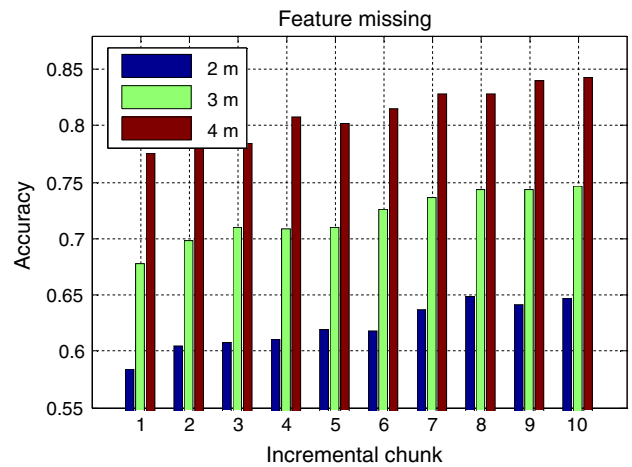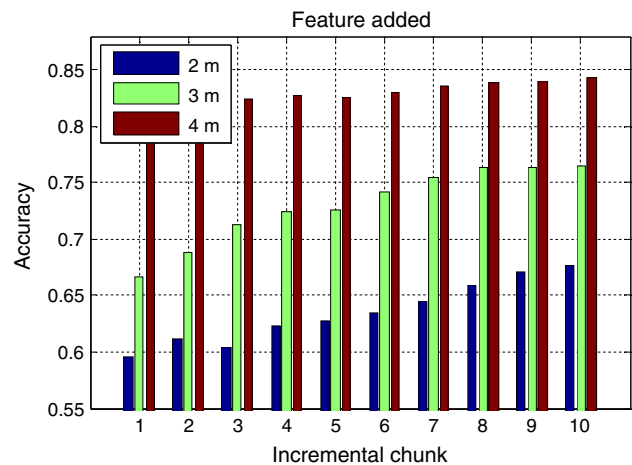

Fig. 16 Accuracy increasing as incremental data comes chunk by chunk in feature increasing case (office area dataset)

somewhere, the location accuracy is not high enough. We can do nothing but to recollect a batch of new data and retrain a new model. Not only data collection will consume a lot of time and money, but also it is a waste of previous data. So we propose the FA-OSELM method which can use a small amount of data to transfer the original model to a new one. The new model not only retains the original features' characteristic, but also fit for the features' change. All the experiments show that FA-OSELM has better performance than the other methods in all the designed experiments.

References

1. Park MH, KIM HC, Lee SJ (2013) Implementation results and service examples of GPS-Tag for indoor LBS and message service. In: 15th international conference on advanced communication technology (ICACT). IEEE Press, PyeongChang, pp 367–370
2. Liu H, Darabi H, Banerjee P, Liu J (2007) Survey of wireless indoor positioning techniques and systems. IEEE Trans Syst Man Cybern Part C 37(6):1067–1080
3. Kjægaard MB (2007) A taxonomy for radio location fingerprinting. In: Hightower J, Schiele B, Strang T (eds) Location- and context-awareness, LNCS, vol 4718. Springer, Berlin, pp 139–156
4. Brunato M, Battiti R (2005) Statistical learning theory for location fingerprinting in wireless LANs. Comput Netw 47:825–845
5. Bahl P, Padmanabhan (2000) RADAR: an in-building RF-based user location and tracking system. In: Proceeding of INFOCOM 2000. IEEE Press, Tel Aviv, pp 775–784
6. Yim J (2008) Introducing a decision tree-based indoor positioning technique. Expert Syst Appl 34(2):1296–1302
7. Ito S, Kawaguchi N (2005) Bayesian based location estimation system using wireless LAN. In: PerCom 2005 workshops. IEEE Press, Kauai Island, pp 273–278
8. Ahmad U, Nasir U, Iqbal M (2006) In-building localization using neural networks. In: IEEE international conference on engineering of intelligent systems. IEEE Press, Islamabad, pp 1–6
9. Huang GB, Zhu QY, Siew CK (2004) Extreme learning machine: a new learning scheme of feedforward neural networks. In: 2004 international joint conference on neural networks (IJCNN'2004), vol 2. IEEE Press, Budapest, pp 985–990
10. Chandra R, Mahajan R, Moscibroda T (2008) A case for adapting channel width in wireless networks. In: Proceedings of the ACM SIGCOMM 2008 conference on data communication, vol 38. ACM Press, New York, pp 135–146
11. Chen Y, Yang Q, Yin J, Chai X (2006) Power-efficient access-point selection for indoor location estimation. IEEE Trans Knowl Data Eng 18:877–888
12. Roos T, Myllymäki P, Tirri H (2002) A probabilistic approach to WLAN user location estimation. Int J Wirel Inf Netw 9:155–164
13. Wu CL, Fu LC, Lian FL (2004) Wlan location determination in ehome via support vector classification. In: IEEE international conference on networking, sensing and control, vol 2. IEEE Press, Chicago, pp 1026–1031
14. Huang GB, Siew CK (2004) Extreme learning machine: RBF network case. In: Proceedings of the eighth international conference on control, automation, robotics and vision, vol 2. IEEE Press, Kunming, pp 1029–1036
15. Huang GB, Siew CK (2005) Extreme learning machine with randomly assigned RBF kernels. Int J Inf Technol 11:16–24
16. Liang NY, Huang GB, Saratchandran P, Sundararajan N (2006) A fast and accurate on-line sequential learning algorithm for feedforward networks. IEEE Trans Neural Netw 17:1411–1423
17. Huang G-B, Chen L, Siew C-K (2006) Universal approximation using incremental constructive feedforward networks with random hidden nodes. IEEE Trans Neural Netw 17(4):879–892
18. Huang G-B, Chen L (2007) Convex incremental extreme learning machine. Neurocomputing 70(16–18):3056–3062
19. Huang G-B, Chen L (2008) Enhanced random search based incremental extreme learning machine. Neurocomputing 71(16–18):3460–3468
20. Huang G-B et al (2012) Extreme learning machine for regression and multiclass classification. IEEE Trans Syst Man Cybern Part B Cybern 42(2):513–529
21. Andrés BC, Pedro JGL, José-Luis SG (2013) Neural architecture design based on extreme learning machine. Neural Netw 48:19–24
22. Wang X, Shao Q, Qi M, Zhai J (2013) Architecture selection for networks trained with extreme learning machine using localized generalization error model. Neurocomputing 102:1–9
23. Miche Y, Heeswijk M, Bas P, Simula O, Lendasse A (2011) TROP-ELM: a double-regularized ELM using LARS and Tikhonov regularization. Neurocomputing 74:2413–2421
24. Zhai J, Xu H, Li Y (2013) Fusion of extreme learning machine with fuzzy integral. Int J Uncertain Fuzziness Knowl Based Syst 21(Suppl. 2):23–34
25. Zhai J, Xu H, Wang X (2012) Dynamic ensemble extreme learning machine based on sample entropy. Soft Comput 16(9):1493–1502
26. Liu J, Chen Y, Liu M, Zhao Z (2011) SELM: semi-supervised ELM with application in sparse calibrated location estimation. Neurocomputing 74(16):2566–2572
27. Liu J, Yang G, Chen Y, Cao Y (2013) Incremental localization in WLAN environment with timeless management. Chin J Comput 36(7):1448–1455
28. Zhao Z, Chen Z, Chen Y, Wang S, Wang H (2014) A class incremental extreme learning machine for activity recognition. Cognit Comput 6(3):423–431
29. Chen Y, Zhao Z, Wang S, Chen Z (2012) Extreme learning machine based device displacement free activity recognition model. Soft Comput 16(9):1617–1625
30. Chen Z, Wang S, Shen Z, Chen Y, Zhao Z (2013) Online sequential ELM based transfer learning for transportation mode recognition. In: The 6th IEEE international conference on cybernetics and intelligent systems (CIS 2013), pp 78–83
31. Wang S, Chen Y, Chen Z (2013) Recognizing transportation mode on mobile phone using probability fusion of extreme learning machines. Int J Uncertain Fuzziness Knowl Based Syst (IJUFKS) 21(Suppl 02):13–22
32. Chen Z, Chen Y, Hu L, Wang S, Jiang X, Ma X, Lane ND, Campbell AT (2014) ContextSense: unobtrusive discovery of incremental social context using dynamic bluetooth data. In: The 2014 ACM international joint conference on pervasive and ubiquitous computing (Ubicomp2014), pp 23–26
33. Serre D (2002) Matrices: theory and applications. Springer, New York
34. Rao CR, Mitra SK (1971) Generalized inverse of matrices and its applications. Wiley, New York
35. Xiao W, Liu P, Soh WS, Jin Y (2012) Extreme learning machine for wireless indoor localization. In: Proceedings of the 11th international conference on information processing in sensor networks. ACM Press, New York, pp 101–102
36. UCI Machine Learning Repository. http://archive.ics.uci.edu/ml