IEEE Access
Multidisciplinary : Rapid Review : Open Access Journal

Received December 4, 2017, accepted January 16, 2018, date of publication January 29, 2018, date of current version March 9, 2018.

Digital Object Identifier 10.1109/ACCESS.2018.2795738

Multi-Device Fusion for Enhanced Contextual Awareness of Localization in Indoor Environments

YALI YUAN[1], CHRISTIAN MELCHING[1], YACHAO YUAN[2], AND DIETER HOGREFE[1]
[1]Telematics Group, Institute of Computer Science, University of Göttingen, 37077 Göttingen, Germany
[2]Sustainable Mobility Research Group, Faculty of Economic Sciences, University of Göttingen, 37073 Göttingen, Germany

Corresponding author: Yali Yuan (yali.yuan@informatik.uni-goettingen.de)

The work of Y. Yuan and Y. Yuan was supported by the Scholarship from the China Scholarship Council.

ABSTRACT Recently, with various developing sensors, mobile devices have become interesting in the research community for indoor localization. In this paper, we propose Twi-Adaboost, a collaborative indoor localization algorithm with the fusion of internal sensors, such as the accelerometer, gyroscope, and magnetometer from multiple devices. Specifically, the data sets are collected first by one person wearing two devices simultaneously: a smartphone and a smartwatch, each collecting multivariate data represented by their internal parameters in a real environment. Then, we evaluate the data sets from these two devices for their strengths and weaknesses in recognizing the indoor position. Based on that, the Twi-AdaBoost algorithm, an interactive ensemble learning method, is proposed to improve the indoor localization accuracy by fusing the co-occurrence information. The performance of the proposed algorithm is assessed on a real-world dataset. The experiment results demonstrate that Twi-AdaBoost achieves a localization error about 0.39 m on average with a low deployment cost, which outperforms the state-of-the-art indoor localization algorithms.

INDEX TERMS Indoor localization, Twi-AdaBoost, fusion, internal sensors, multiple devices.

I. INTRODUCTION

Typical tasks for indoor localization with mobile devices include many applications, such as medical assistance (patient tracking) [2], elderly care (aged pedestrian tracking) [2] and underground mining safety [4], which have attracted many researchers' attention in recent time. However, it is challenging to obtain the accurate pedestrian localization in indoor environment due to multiple reasons. First of all, it is difficult to measure the distance due to the complexity of human movements in the GPS-denied, crowded and cluttered indoor environment. Also any sensor system used by a pedestrian should be wearable and portable, which makes it difficult to use certain sensors, such as laser range scanners although they can be successfully used in robotic applications [14]. In addition, instead of localizing a target in some area sporadically or on demand, the localization of pedestrian should be continuously and possibly in real-time.

With the availability of new small and inexpensive sensors, which enables practical tracking of individuals (who must carry them at all times), the localization of pedestrian in indoor environment has been improved significantly. In recent years, there has been an increasing interest in the development of pedestrian navigation systems for satellite-denied scenarios. The popularization of smartphones and smartwatches is an interesting opportunity to reduce the infrastructure cost of the positioning systems. If these devices compute their own positions using their internal sensors, it requires very little, if any, physical infrastructure to function. Moreover, this offers a degree of location privacy since users can select whether they share the information with any third party or not.

Some of the existing technological approaches for indoor location systems, such as the infrared light, ultrasonic sensors, WLAN, RFID, Ultra Wideband, ZigBee and computer vision, are not suitable for mobile devices [18] and [19]. Since a dedicated infrastructure or higher processing capabilities are necessary for these technologies, this hinders the systems miniaturization and scalability. In addition, the above technologies can lead to sub-optimal positioning because the communication access points are rarely deployed to provide the optimal location geometry and coverage overlap. Therefore, mobile devices equipped with a variety of sensors (e.g., accelerometer, gyroscope, magnetometer) have become popular in modern indoor localization systems [19].

Motivated by the lack of a comprehensive approach in multi-device based context recognition research, we propose a multi-device context indoor localization system using several kinds of sensors in both smartphone and smartwatch. The Twi-AdaBoost algorithm is utilized to purse the optimal combination of sensors from multiple devices. By following this approach, the system is able to obtain an accurate location. The proposed fusion approach reduces the mean localization errors of position x (0.387m) by 51.26% as compared to using Generalized Regression Neural Network (GRNN) algorithm [25] on the combined dataset, where the datasets of smartphone and smartwatch are merged by simply combining all the features. As for the mean localization error of position y (0.398m), the proposed fusion approach is improved by 62.56% compared to GRNN [25]. The other state-of-the-art indoor localization algorithms, such as Support Vector Regression (SVR) [24] perform worse than GRNN [25] on the simply combined dataset.

The rest of the paper is organized as follows. In Section II, this paper reviews related works. Section III presents the proposed Twi-AdaBoost fusion strategy and its knowledge background. The datasets analysis and preprocessing, experimental results as well as performance evaluations are introduced in Section IV. Finally, Section V concludes the paper.

II. RELATED WORK

In outdoor environments, Global Positioning System (GPS) is one of the most popular way to localize mobile devices. However, in indoor environments where the GPS signals are not receivable or usable, different models were proposed to solve the indoor localization problems.

Chen et al. [28] proposed a Convolutional Neural Network that used the Channel State Information of only one access point and achieved an average localization error of 1.36m, but has a high training complexity. In contrary, a low computational complexity model was proposed in [17] which achieved a localization error of 2.1m. In that model, an AdaBoost algorithm with C4.5 method as a weak classifier was used to combine the RSS and orientation information to improve the accuracy of indoor localization. It included two phases, the offline phase and online phase. In the offline phase, a database of the RSS from different access points at each reference location for the target environment was built; in online phase, the localization was determined by means of a sample of RSS collected in a particular position and an estimation model that used database information. Although the proposed model was not more accurate than other models, it demonstrated that it was possible to execute such models on resource-constrained devices. GRNN was proposed in [25], where RSS data gathered at the access points from the referenced nodes were used to train the GRNN model and the target node position was calculated by the weighted centroid method. Wu et al. [24] used the SVR model to solve the missing value location estimation problem. Utilizing other machine learning technologies, such as LR [27] which is a RSS-based localizaiton method, localization accuracy was improved by correcting the distance circles using LR model.

In [21], a sensor fusion framework was proposed by combining WiFi, Pedestrian Dead Reckoning (PDR) and landmarks. It used the linear Kalman filter to simplify the sensor fusion problem on a smartphone. The weighted path loss algorithm was used in the WiFi localization due to its simplicity and effectivity, while in the pedestrian dead reckoning approach the initial estimation error was amended by landmarks. A Kalman filter was used to fuse magnetometer and gyroscope records in order to improve the accuracy of walking direction estimation. The localization accuracy of this approach was 1m on an average. However, the additional landmarks with the known positions should be provided to help this approach restart when the users went through these landmarks. At the same time, Ma et al. [12] used the weighted fusion to improve the WiFi-based indoor localization. There were two steps in this algorithm: the offline acquisition and the online localization. In the offline acquisition process, the optimal parameters were selected to complete the signal acquisition. In addition, the fingerprints database was built. In the online localization process, a pre-match method was employed to select the candidate fingerprints to shorten the positioning time. Then, two intermediate results were obtained by using the improved Euclidean distance and the improved joint probability. The final results were calculated by fusing these two intermediate results with different weights. More similar work can be found in [9]–[11]. However, the time required to install, configure and maintain the WiFi systems together with the expense of access points have limited the general deployment of these indoor algorithms.

Fusing the internal sensors is popular in human activity recognition [13], [16]. For example, in [16], the data coming from embedded sensors on the smartphone and environmental sensors were fused by a decision tree based on multi-sensor data-stream. Then they used the Recurrent Neural Networks (RNNs) to model the RSS stream.

However, few previous researchers did work on sensor fusion from multi-device in indoor localization. In this paper, to use the richer context information, we propose a Twi-AdaBoost algorithm which combines the data of self-contained sensors from multiple devices, like smartphone and smartwatch.

III. METHODOLOGY

Indoor localization has been an important issue in recent time. To solve this problem, a Twi-AdaBoost fusion strategy is proposed, exploiting the intrinsic correlation between two conditional independent datasets from smartphone and smartwatch to boost the ability of prediction of the pedestrian's location from a crowded and cluttered background.

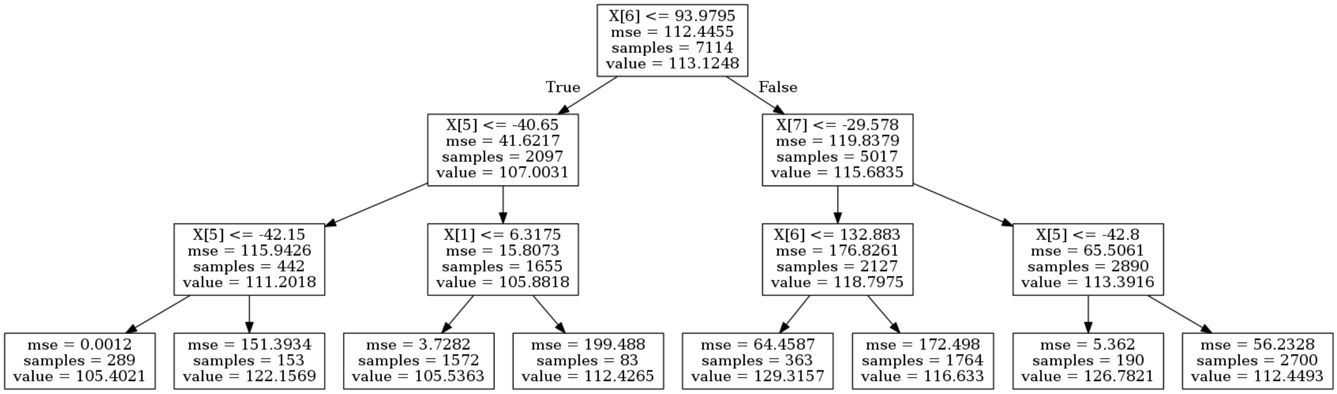

FIGURE 1. Example of cart regression tree.

A. BUILDING A WEAK LEARNER BASED ON THE CART ALGORITHM

In 1984, Breiman et al. [15] proposed the CART method by building a binary decision tree according to some splitting rules based on the predictor variables to address the regression problem. Suppose that the CART method takes a training dataset with instances $(x_1, y_1), \cdots, (x_n, y_n)$ as input, where each x_i belongs to the features space X (such as accelerometer, gyroscope and magnetometer) and each label y_i is in the reference location dataset and $y_i \in R$. Fig. 1 depicts an example of the cart regression tree based on the experiment dataset [6]. The subsets created by the splits are named nodes, otherwise, they will be named by terminal nodes. A regression tree partitions the X-space into disjoint regions A_k and provides a fitted value $E(Y|X \in A_k)$ within each region.

The tree is implemented recursively with the following steps in Algorithm 1.

Algorithm 1 Construction of Cart regression model

Input : Training dataset $(x_1, y_1), \cdots, (x_n, y_n)$ *and* $y \in R$; *CART*.

for *Each node* $x_i \in X$ **do**

  Examine every allowable split on each reference location variable y_i. Binary questions, like *Is* $x_i > c$?, are used to generate the binary splits.
  Select and execute the 'best' of these splits.
  Stop splitting on a node when some stopping rule is reached.

end

Output: CART regression model $H(x)$

The CART regression method is selected as the weak leaner based on the following two main reasons:

- It is simple and fast. In addition, it is not significantly impacted by outliers in the input variables.
- It is nonparametric and does not rely on the dataset distribution.

B. ADABOOST.R2 REGRESSION MODEL

AdaBoost.R2 is one of most popular ensemble learning algorithms, which is designed to solve the regression problem [5]. In AdaBoost.R2, a set of weak classifiers are trained to form a strong classifier. Initially, each training instance receives a uniform weight w_i, which indicates the relative importance of each instance. After each iteration, the weight of the instance with the larger real-valued error $e_i = \frac{|y_i - h_t(x_i)|}{max_{i=0}^{n}|e_i|}$ will be increased, otherwise, the weight will be decreased. In this case, the weaker learner is forced to focus on the "hard" examples in the training dataset. In particular, three loss functions can be selected in AdaBoost.R2: $e_i' = \frac{e_i}{D}$ (*linear*), $e_i' = (\frac{e_i}{D})^2$ (*square*), $e_i' = 1 - exp(-e_i/D)$ (*exponential*). The pseudo code of AdaBoost.R2 is given in Algorithm 2.

C. TWI-ADABOOST FUSION STRATEGY

Localization techniques based on individual dataset have their own strengths and weaknesses. In this paper, we investigate the potential of fusing both smartphone and smartwatch datasets to better infer the pedestrian's indoor localization.

Fig. 2 depicts the proposed Twi-AdaBoost algorithm based on the collaborative exploitation of smartphone-smartwatch characteristics. The training datasets are first extracted from smartphone and smartwatch using their internal sensors, such as the accelerometer, gyroscope and magnetometer. Then, the Twi-AdaBoost strategy is used to improve the localization performance. Ultimately, the accurate location is obtained by combing all generated weak learners.

Fig. 3 illustrates the interactive ensemble learning process across multiple datasets to form a consolidated fusion by interactively exploiting the complementary sensor features from different devices, which is the key difference from the traditional AdaBoost.R2 algorithm. The pseudo-code for Twi-AdaBoost is given in Algorithm 3. Twi-AdaBoost works by training the weak learner with an initial sample weight and evaluating its prediction by comparing the results to each other in form of the penalty factor. With this information new weights are generated and used for the next iteration.

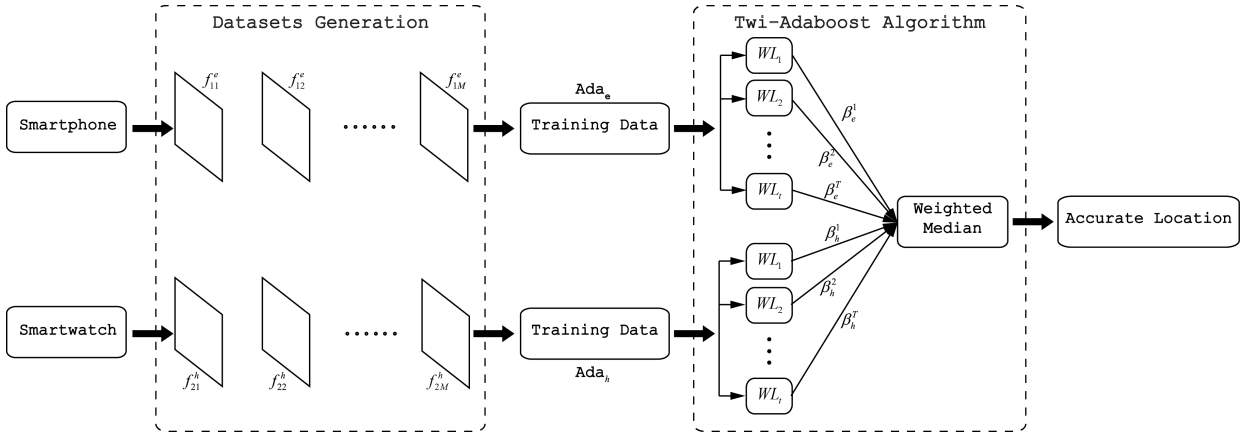

FIGURE 2. Proposed Twi-AdaBoost algorithm based on the collaborative exploitation of smartphone and smartwatch.

Algorithm 2 AdaBoost.R2

Input : Training dataset $(x_1, y_1), \ldots, (x_n, y_n)$ $y \in R$; *WeakLearner*; Iteration T; Initial weight distribution $D_t i = \frac{1}{n}$, $i \in [1, n]$

for *each iteration* $t \in [1, T]$ **do**

Call *WeakLearner*, providing it with a distribution D_t.

Build the regression model: $h_t(x) \to y$ for regression problems.

for *each instance* x_i **do**

Calculate the adjusted error $e_i^t = \frac{1-exp(-|y_i - h_t(x_i)|)}{D_t}$. $D_t = max_{j=1}^{n} |y_j - h_t(x_j)|$.

end

Calculate the adjusted error of h_t: $\epsilon_t = \sum_{i=1}^{n} e_i^t w_i^t$; if $\epsilon_t \geq 0.5$, stop and set $N = t - 1$.

$\beta_t = \frac{\epsilon_t}{1-\epsilon_t}$.

Update the weight vector: $w_i^{t+1} = \frac{w_i^t \beta_t^{1-e_i^t}}{Z_t}$, where Z_t is a normalization factor selected such that w_i^{t+1} will be a distribution.

end

Output: Strong classifier $H(x)$ is the weighted median of $h_t(x)$ for $t \in [1, T]$, using $log\frac{1}{\beta_t}$ as the weight.

In Algorithm 3, initially, each sample has a uniform weight $D_j^1(i) = 1/M$ of the ith training sample on the jth dataset, which indicates the relative contribution of each sample for the final prediction result. The weight will be changed after each iteration. The weight $D_j^{t+1}(x_{ji})$ of each sample in Twi-AdaBoost is decided by both the real-valued error $L_j^t(x_{ji})$ and the punishment factor $P_j^t(x_{ji})$, which is introduced to convey the complementary characteristics across the two datasets into the ensemble learning process. The penalty degree of the weight is controlled by the scale factor $P_j^t(x_{ji})$, which is decided by the value of $p_j^t(x_{ji})$ of all weak learners $f_j^t(x_{ji})$ achieving the agreement with both $f_k^t(x_{ji})$ and y_i at the tth iteration. With exploitation of both datasets from smartphone and smartwatch, the ''hardest'' samples will be punished with the largest weights, which forces the new weak learners to focus on the ''hardest'' samples in the next generation and helps this algorithm to achieve better performance.

IV. EXPERIMENTS

A. DATASET ANALYSIS

In the experiment, the indoor localization datasets of paper [6] are employed to test the proposed algorithm. The datasets with over 36000 continuous samples are collected in a 185.12 m^2 real indoor environment. The user was wearing two devices simultaneously, such as a Sony Xperia M2 smartphone and a LG W110G smartwatch, to collect the data in each campaign. Fig. 4 from paper [6] depicts the overall map, where the data collection was performed. There are two rooms, two corridors and one small entrance hall inside this indoor office environment. Each dot in the map corresponds to a detection point and each dot is 0.6 meters far from another since each dot occupies 0.6 m × 0.6 m. For each of them, features of sensors in each device were collected. A zig-zag trajectory was performed by two different users who were wearing the same equipments to cover the entire map. The walking speed of each user was 0.6 m/s on an average. Each sample was collected about every 100 millisecond and the collection time is very short.

All the recorded datasets include the following features:

- Place ID, Timestamp;
- Accelerometer_X, Accelerometer_Y, Accelerometer_Z, MagneticField_X, MagneticField_Y, MagneticField_Z, X_Axis Angle (Pitch), Y_Axis Angle (Roll), Z_Axis Angle (Azimuth), Gyroscope_X, Gyroscope_Y, Gyroscope_Z.

The exact numbers of recorded samples in each measurement can be found in table 1.

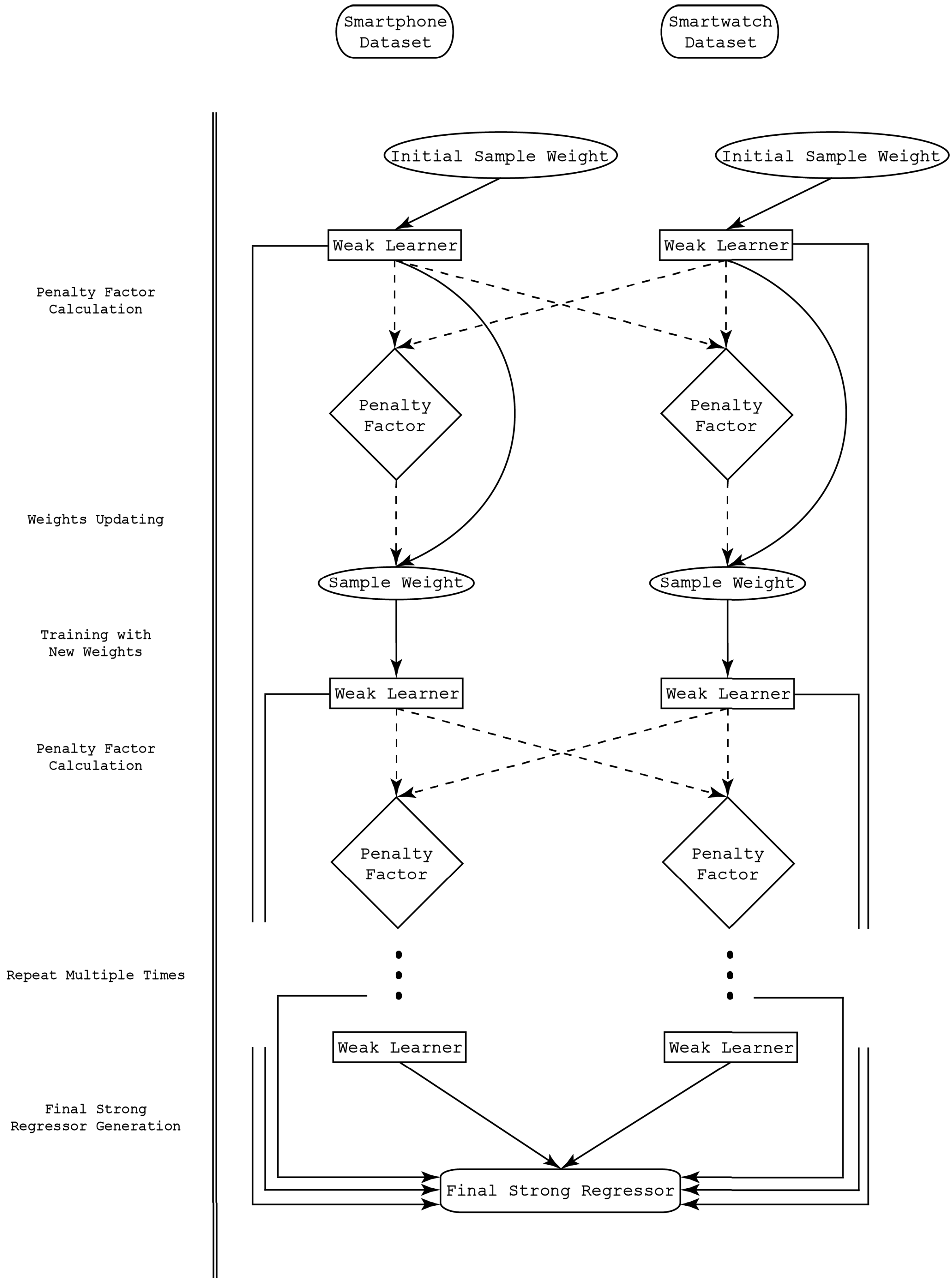

FIGURE 3. Interactive training process of Twi-AdaBoost across datasets from both smartphone and smartwatch.

B. DATASET PREPROCESSING

The datasets [6] were collected by recording the internal sensor data of different devices about every 100 millisecond when the walking speed of each user was 0.6 m/s on an average. Thus, they might be not perfectly synchronized and have a slight offset in time. Therefore, we filter the provided

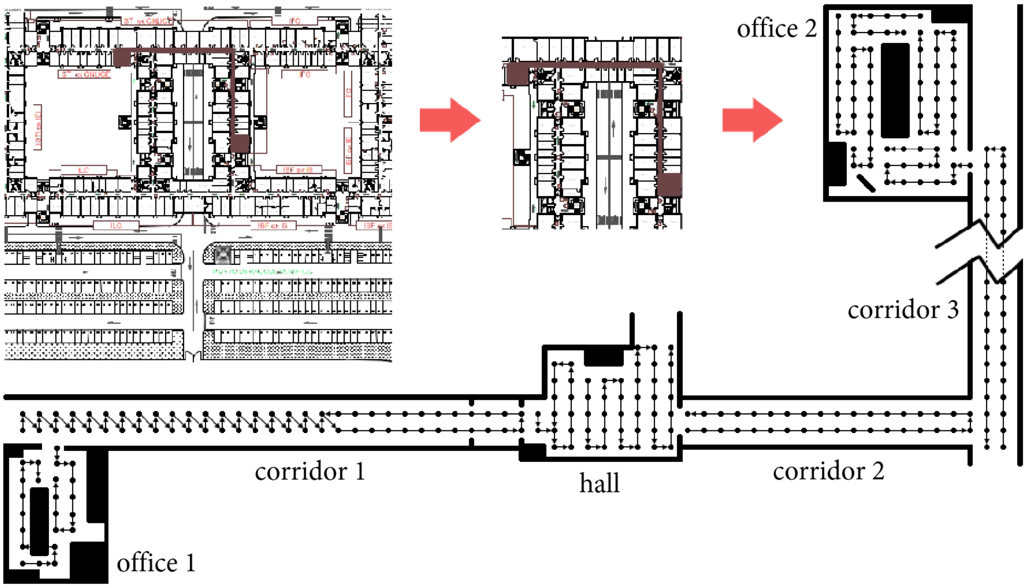

FIGURE 4. Map of the data collecting environment.

TABLE 1. Number of total samples recorded.

	Smartphone	Smartwatch
Measurement 1	18355	58370
Measurement 2	17787	58370

data beforehand to enable their suitable for our algorithm. Furthermore, some samples are not usable, as their precise recording position is unknown.

The samples of datasets are preprocessed and filtered out according to the following conditions:

1) As the datasets contain recorded samples that are not uniquely assignable to the given reference points, some samples are removed in order to assure the correct labeling of the data.
2) Each sample needs to have a counterpart-sample in all other datasets, which was recorded within a 50ms sliding window, in order to make sure that the sample-pairs were recorded almost simultaneously. For example, a sample pair consists of two samples, one recorded on the smartphone and one recorded on the smartwatch with max time difference of 50ms.
3) Each sample can only be chosen either once or never to ensure that no sample is used twice and therefore unintentionally weighted higher than the other ones. This creates a one-to-one relationship between the selected samples of each dataset, illustrated in Fig. 5.

After filtering out the datasets according to the above conditions, there are 14228 samples and 12608 samples from both smartphone and smartwatch in the first measurement and second measurement, respectively.

Finally, all features are normalized using min-max scaling technique [23]. The definition of it is given in equation (1), where $x \in X$ is the original and $x' \in [a, b]$ is the rescaled value.

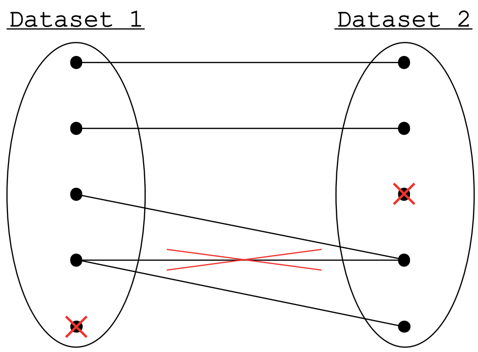

FIGURE 5. Illustration of a valid possible connection between samples of each dataset after preprocessing.

$$x' = \frac{(b-a)(x - min(X))}{max(X) - min(X)} \tag{1}$$

C. PERFORMANCE METRICS

In order to evaluate the results of Twi-AdaBoost algorithm, the performance metrics provided by scikit-learn [22] are employed in this paper. For example Root Mean Squared Error (RMSE), which is used to measure the differences between the values estimated by a model and the values actually observed; Explained Variance Score (EVS), which is used to compute the explained variance regression score; Mean Absolute Error (MAE), which is a risk metric corresponding to the expected value of the absolute error loss as well as the box-and-whiskers plots, which can display the variation in samples of a statistical population and detect the outliers being plotted as individual points.

The RMSE estimated over $n_{samples}$ is defined as equation (2), where y'_i is the predicted value of the i-th sample

Algorithm 3 Multi-Device AdaBoost Algorithm

Input : The training dataset $S = \{(x_{ji}, y_i)_{j=1,\ldots,N;\ i=1,\ldots,M}\}$, where N is the number of different datasets from different devices and M the number of samples; *WeakLearner*; Iteration T; Initial weight of each sample: $D_j^1(i) = 1/M$.

for $t = 1$ *to* T **do**

  for $j = 1$ *to* N **do**

    Get a random integer $r \in [1, M]$ and generate a subset R, containing the r highest weighted samples of S.

    Train the weak classifier with R and D_j^t and build the regression model $f_j^t(x_j)$.

  end

  for $j = 1$ *to* N **do**

    Calculate the distance of each sample x_{ji} in S and the prediction with $l_j^t(x_{ji}) = |f_j^t(x_{ji}) - y_i|$.

    Calculate the loss function $L_j^t(x_{ji})$ for each sample using the exponential loss function as $L_j^t(x_{ji}) = 1 - exp(-\frac{l_j^t(x_{ji})}{max_{i=1,\ldots,M}(l_j^t(x_{ji}))})$.

    Calculate the weighted loss as $\bar{L}_j^t = \sum_{i=1}^{M} L_j^t(x_{ji}) D_j^t(x_{ji})$.

    Set $\beta_j^t = \frac{\bar{L}_j^t}{1-\bar{L}_j^t}$.

    For each sample x_{ji} in S, calculate the punishment factor $P_j^t(x_{ji}) = 1 - exp(-\frac{p_j^t(x_{ji})}{max_{i=1,\ldots,M}(p_j^t(x_{ji}))})$, where $p_j^t(x_{ji}) = \frac{1}{N}(|f_j^t(x_{ji}) - y_i| + \sum_{k=1}^{N}(|f_j^t(x_{ji}) - f_k^t(x_{ji})|)$.

    For each sample x_{ji} in S, set $D_j^{t+1}(x_{ji}) = \frac{D_j^t(x_{ji})\beta_t^{(1-L_j^t(x_{ji}))(1-P_j^t(x_{ki}))}}{Z^t}$ where Z_j^t is the normalization factor such that D_j^{t+1} will be a distribution.

  end

end

Output: The strong classifier $F(x)$ is the weighted median of $f_j^t(x_j)_{(t=1,\ldots,T;\ j=1,\ldots,N)}$, with $log(\frac{1}{\beta_j^t})$ used as the weight.

and y_i is the corresponding true value. The smaller the RMSE value is, the better the performance of the proposed Twi-AdaBoost algorithm. The EVS is estimated as equation (3), where *Var* is the variance, i.e. the square of the standard deviation. The higher the value is, the better the performance. The best possible score is 1.0. Equation (4) depicts the MAE estimated over $n_{samples}$.

$$RMSE(y, y\prime) = \sqrt{\frac{1}{n_{samples}} \sum_{i=0}^{n_{samples}-1} (y_i - y_i')^2} \qquad (2)$$

$$EVS(y, y\prime) = 1 - \frac{Var\{y - y\prime\}}{Var\{y\}} \qquad (3)$$

$$MAE(y, y\prime) = \frac{1}{n_{samples}} \sum_{i=0}^{n_{samples}-1} (|y_i - y_i'|) \qquad (4)$$

D. GENERAL RESULTS AND ANALYSIS

To verify the performance of the proposed Twi-AdaBoost algorithm, we use 12608 samples including 6304 samples of smartphone and 6304 samples of smartwatch in the second measurement. About 85% samples are randomly selected as the training set and the rest samples as testing set. The metrics RMSE, EVS as well as box-and-whiskers plots are utilized to evaluate the performance. In all the figures, AdaBoost.R2 on SH denotes AdaBoost.R2 on smartphone dataset; AdaBoost.R2 on SW denotes AdaBoost.R2 on smartwatch dataset while AdaBoost.R2 on HW denotes AdaBoost.R2 on the mixed dataset, where the datasets of smartphone and smartwatch are merged by simply combining all the features.

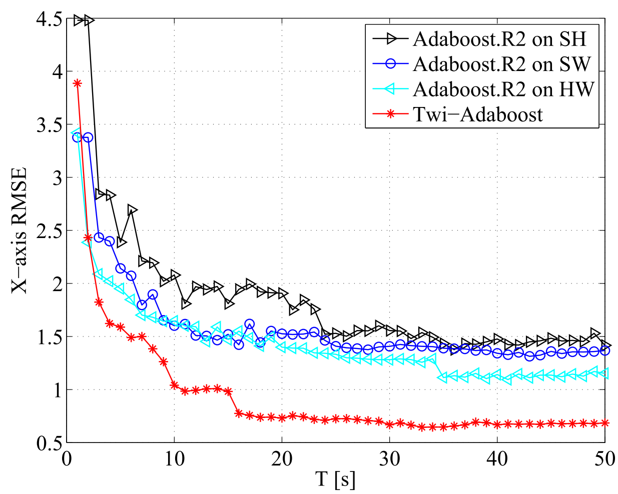

FIGURE 6. RMSE of position x estimation using Twi-AdaBoost and AdaBoost.R2 methods.

1) ROOT MEAN SQUARE ERROR

In both Fig. 6 and Fig. 7, we can see that with the iteration increase, the RMSE decreases. At the 50th iteration, both AdaBoost.R2 and Twi-AdaBoost tend to be stable. Fig. 6 depicts the RMSE of position x estimation using Twi-AdaBoost and AdaBoost.R2 methods with iteration T on the testing set. It is clear from Fig. 6 that compared to AdaBoost.R2 on SH (1.42), the RMSE value achieved by Twi-AdaBoost (0.69) is 51.63% lower while the RMSE value obtained by Twi-AdaBoost is about 49.90% lower than that of AdaBoost.R2 on SW (1.37) at the 50th iteration. Compared with AdaBoost.R2 on HW, Twi-AdaBoost achieves 40.54% improvement. Fig. 7 shows that RMSE of position y estimation using Twi-AdaBoost and AdaBoost.R2 methods with iteration T on the testing set. We notice that the RMSE of AdaBoost.R2 on SW (1.26) is better than that of AdaBoost.R2 on SH (1.81). However, the RMSE value

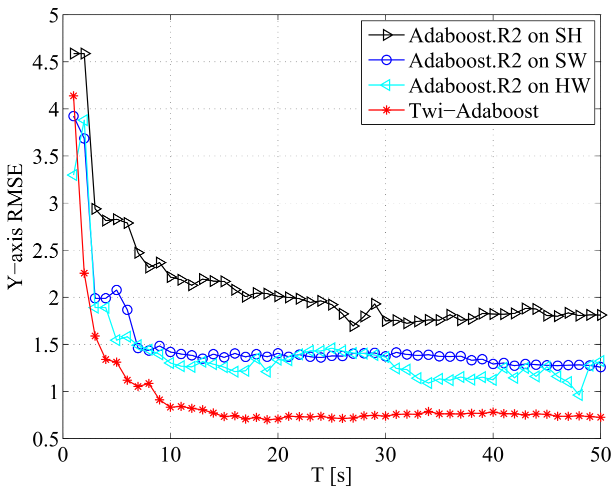

FIGURE 7. **RMSE of position *y* estimation using Twi-AdaBoost and AdaBoost.R2 methods.**

achieved by Twi-AdaBoost (0.73) is 42.32% lower compared to AdaBoost.R2 on SW at the 50th iteration. Compared with AdaBoost.R2 on HW (1.32), Twi-AdaBoost achieves 45.10% improvement.

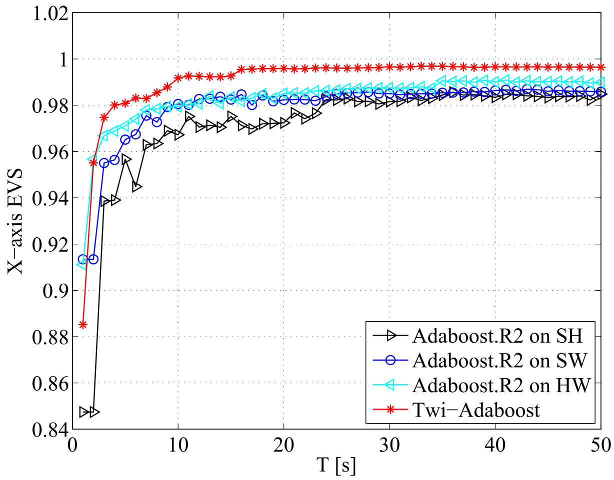

FIGURE 8. **EVS of position *x* estimation using Twi-AdaBoost and AdaBoost.R2 methods.**

2) EXPLAINED VARIANCE SCORE

As visible in Fig. 8 and Fig. 9, the EVS of both position x and y increase with the increase of iteration. However, after the 30th iteration, the performance of EVS becomes stable. Fig. 8 describes that the EVS of position x estimation using Twi-AdaBoost and AdaBoost.R2 methods with iteration T on the testing set. It is demonstrated that Twi-AdaBoost outperforms AdaBoost.R2 on SH, SW and HW, respectively. Specifically, the EVS of position x estimation of it attains 1.19%, 1.08% and 0.66% higher compared to AdaBoost.R2 on SH, AdaBoost.R2 on SW and AdaBoost.R2 on HW, respectively. In Fig. 9, it shows the EVS of position y estimation using Twi-AdaBoost and AdaBoost.R2 methods with iteration T on the testing set. The EVS of position y estimation of

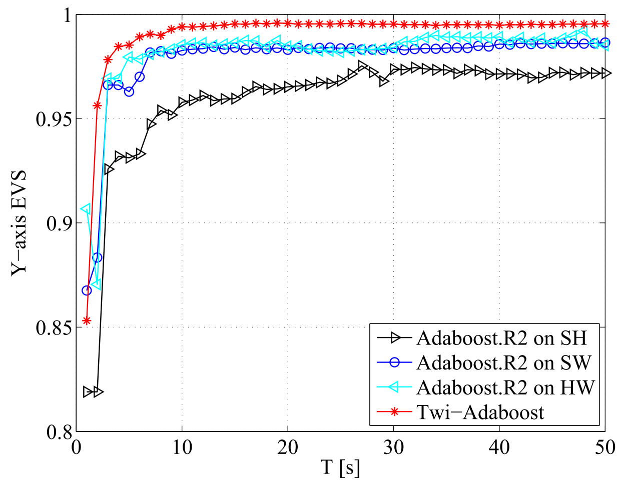

FIGURE 9. **EVS of position *y* estimation using Twi-AdaBoost and AdaBoost.R2 methods.**

AdaBoost.R2 on SH is worse than that of AdaBoost.R2 SW while the EVS of AdaBoost.R2 on HW has almost same performance with that of AdaBoost.R2 SW.The EVS of y position estimation attained by Twi-AdaBoost is the highest one with the EVS value 99.55%.

3) BOX-AND-WHISKERS PLOTS

Box-and-whiskers plots of the predicted position offsets are shown in Fig. 10, Fig. 12 and Fig. 13, which is a more complete performance analysis. The boxes refer to different values of the updated period T based on different datasets using AdaBoost.R2 and Twi-AdaBoost, where the boundaries of the box represent the 25th and 75th percentiles of the sample data, respectively; the line within the box shows the median; Whiskers above and below the box indicate the range from the 90th percentiles and 10th percentiles, respectively; the outliers are shown as dots. Notice that there are more outliers in Fig. 10 and Fig. 12, which are obtained from smartphone and smartwatch dataset using AdaBoost.R2, respectively. We can see that Fig. 13 obtains the best performance.

TABLE 2. **Comparison results among Twi-AdaBoost and the state-of-the-art.**

	XRE	YRE	XMAE	YMAE	XEVS	YEVS
Twi-Ada	**1.073**	**0.824**	**0.387**	**0.398**	**0.991**	**0.994**
LR [27]	8.658	9.123	6.746	7.729	0.411	0.286
Ada.RT [26]	2.375	2.020	0.862	0.975	0.955	0.965
GRNN [25]	1.905	2.455	0.794	1.063	0.971	0.948
SVR [24]	7.328	7.908	5.071	6.475	0.584	0.464

E. COMPARISON RESULTS AND ANALYSIS

Table 2 illustrates the comparison results among the proposed Twi-AdaBoost method and the state-of-the-art indoor localization algorithms, where XRE denotes the RMSE on X coordinate; YRE denotes the RMSE on Y coordinates; Twi-Ada is Twi-AdaBoost; Ada.RT is AdaBoost.RT. It displays the performance of the different models, using the metrics introduced previously, given the HW dataset, where

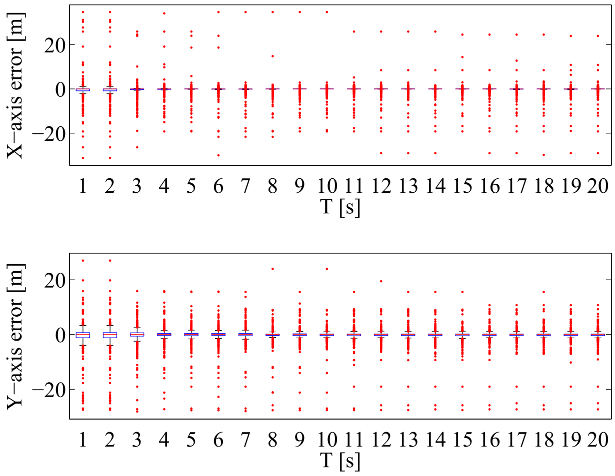

FIGURE 10. **Box-and-whiskers plots of the position offsets using AdaBoost.R2 based on the smartphone dataset.**

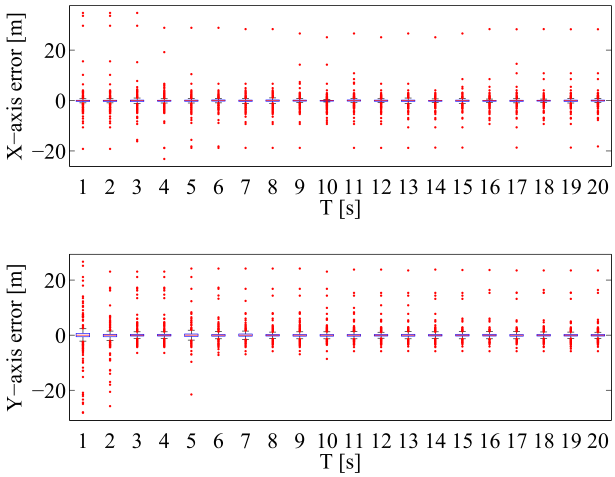

FIGURE 11. **Box-and-whiskers plots of the position offsets using AdaBoost.R2 based on the smartwatch dataset.**

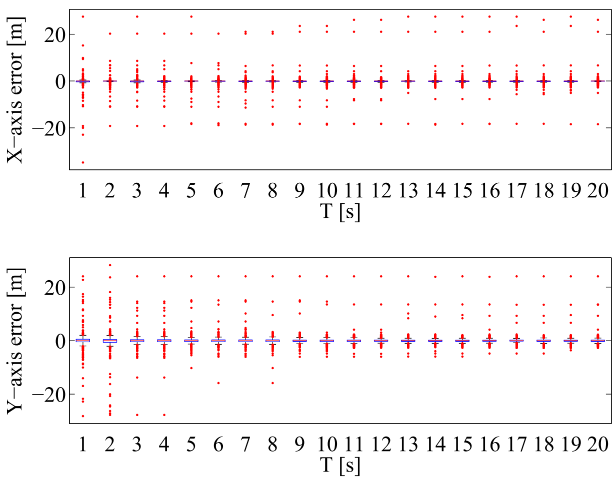

FIGURE 12. **Box-and-whiskers plots of the position offsets using AdaBoost.R2 based on the mixed dataset.**

the datasets of smartphone and smartwatch are merged by simply combining all the features. We conduct extensive numerical studies on randomly selected different initial data.

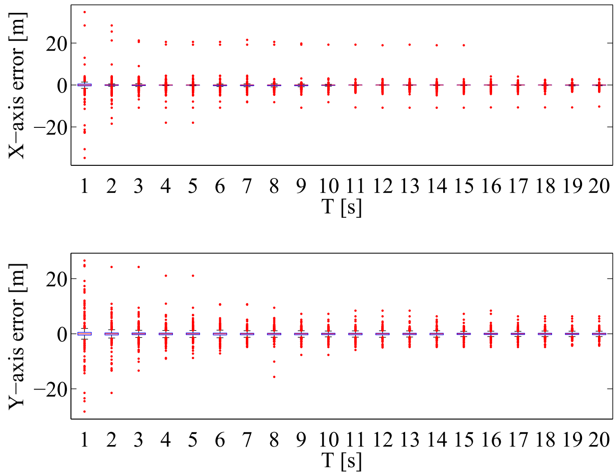

FIGURE 13. **Box-and-whiskers plots of the position offsets using Twi-AdaBoost based on the mixed dataset.**

It is clear to see that the proposed Twi-AdaBoost outperforms the other algorithms throughout all metrics on both coordinates. GRNN is perfomring second best on the X coordinate, but worse than Ada.RT on the Y coordinate.

V. CONCLUSION

In this paper, we introduce Twi-AdaBoost, an indoor collaborative localization algorithm that explores the accelerometer, gyroscope and magnetometer sensors on both smartphone and smartwatch. The key contribution of the proposed Twi-AdaBoost algorithm is fusing the co-occurrence information to get a better performance for the indoor localization based on the real world data. The indoor localization datasets [6] used in this paper have the multisource characteristics, which are supported by the presence of two different devices collecting data simultaneously from the surrounding environment: a smartphone and a smartwatch, respectively. Each device collects multivariate data represented by their internal sensors, such as acceleration, orientation, and gyroscope. From the experiment results, it is obvious that Twi-AdaBoost convincingly outperforms the state-of-the-art indoor localization algorithms, taking advantage of the co-occurrence correlation across the sensors from multiple devices. Specifically, the localization error of position x and y achieved by Twi-AdaBoost is 0.387m and 0.398m, respectively.

Considering the future work, we plan to utilize the correlation between the position x and y in the same location to improve the performance of the indoor localization in this paper. In addition, we will focus on exploiting the datasets combined by more co-occurrence information from multiple devices, like the Camera and WiFi, by machine learning methods to improve the localization accuracy in indoor environment.

ACKNOWLEDGMENT

The authors are grateful to Dr. Francesco Piccialli for his help and support. The authors would also like to thank the referees' comments which have been incorporated into the paper.

REFERENCES

[1] A. Colombo, D. Fontanelli, D. Macii, and L. Palopoli, "Flexible indoor localization and tracking based on a wearable platform and sensor data fusion," *IEEE Trans. Instrum. Meas.*, vol. 63, no. 4, pp. 864–876, Apr. 2014.

[2] U. Varshney, "Pervasive healthcare and wireless health monitoring," *Mobile Netw. Appl.*, vol. 12, nos. 2–3, pp. 113–127, 2007.

[3] J.-Y. Wang, C.-P. Chen, T.-S. Lin, C.-L. Chuang, T.-Y. Lai, and J.-A. Jiang, "High-precision RSSI-based indoor localization using a transmission power adjustment strategy for wireless sensor networks," in *Proc. IEEE 14th Int. Conf. High Perform. Comput. Commun. (HPCC-ICESS)*, Jun. 2012, pp. 1634–1638.

[4] A. Chehri, P. Fortier, and P. Tardif, "UWB-based sensor networks for localization in mining environments," *Ad Hoc Netw.*, vol. 7, no. 5, pp. 987–1000, 2009.

[5] H. Drucker, "Improving regressors using boosting techniques," in *Proc. ICML*, vol. 97. 1997, pp. 107–115.

[6] P. Barsocchi, A. Crivello, D. La Rosa, and F. Palumbo, "A multisource and multivariate dataset for indoor localization methods based on WLAN and geo-magnetic field fingerprinting," in *Proc. Int. Conf. Indoor Positioning Indoor Navigat. (IPIN)*, 2016, pp. 1–8.

[7] H. Qi and J. B. Moore, "Direct Kalman filtering approach for GPS/INS integration," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 38, no. 2, pp. 687–693, Apr. 2002.

[8] Q. Li, W. Li, W. Sun, J. Li, and Z. Liu, "Fingerprint and assistant nodes based Wi-Fi localization in complex indoor environment," *IEEE Access*, vol. 4, pp. 2993–3004, 2016.

[9] S. Hilsenbeck, D. Bobkov, G. Schroth, R. Huitl, and E. Steinbach, "Graph-based data fusion of pedometer and WiFi measurements for mobile indoor positioning," in *Proc. ACM Int. Joint Conf. Pervasive Ubiquitous Comput.*, 2014, pp. 147–158.

[10] C. Wu, Z. Yang, Z. Zhou, Y. Liu, and M. Liu, "Mitigating large errors in WiFi-based indoor localization for smartphones," *IEEE Trans. Veh. Technol.*, vol. 66, no. 7, pp. 6246–6257, Jul. 2017.

[11] G. Fei, J. Niu, and L. Duan, "WAIPO: A fusion-based collaborative indoor localization system on smartphones," *IEEE/ACM Trans. Netw.*, vol. 25, no. 4, pp. 2267–2280, Apr. 2017.

[12] R. Ma, Q. Guo, C. Hu, and J. Xue, "An improved WiFi indoor positioning algorithm by weighted fusion," *Sensors*, vol. 15, no. 9, pp. 21824–21843, 2015.

[13] M. Shoaib, S. Bosch, H. Scholten, P. J. Havinga, and O. D. Incel, "Towards detection of bad habits by fusing smartphone and smartwatch sensors," in *Proc. IEEE Int. Conf. Pervasive Comput. Commun. Workshops (PerCom Workshops)*, Mar. 2015, pp. 591–596.

[14] S. Thrun, W. Burgard, and D. Fox, *Probabilistic Robotics*. London, U.K.: MIT Press, 2005.

[15] L. Breiman, J. Friedman, C. J. Stone, and R. A. Olshen, *Classification and Regression Trees*. Boca Raton, FL, USA: CRC Press, 1984.

[16] F. Palumbo, C. Gallicchio, R. Pucci, and A. Micheli, "Human activity recognition using multisensor data fusion based on reservoir computing," *J. Ambient Intell. Smart Environ.*, vol. 8, no. 2, pp. 87–107, 2016.

[17] D. Sanchez-Rodriguez, P. Hernandez-Morera, J. M. Quinteiro, and I. Alonso-Gonzalez, "A low complexity system based on multiple weighted decision trees for indoor localization," *Sensors*, vol. 15, no. 6, pp. 14809–14829, 2014.

[18] A. K. M. M. Hossain and W.-S. Soh, "A survey of calibration-free indoor positioning systems," *Comput. Commun.*, vol. 66, pp. 1–13 Jul. 2015.

[19] R. Harle, "A survey of indoor inertial positioning systems for pedestrians," *IEEE Commun. Surveys Tuts.*, vol. 15, no. 3, pp. 1281–1293, 3rd Quart., 2013.

[20] Y. Chen, Q. Yang, J. Yin, and X. Chai, "Power-efficient access-point selection for indoor location estimation," *IEEE Trans. Knowl. Data Eng.*, vol. 18, no. 7, pp. 877–888, Jul. 2006.

[21] Z. Chen, H. Zou, H. Jiang, Q. Zhu, Y. C. Soh, and L. Xie, "Fusion of WiFi, smartphone sensors and landmarks using the Kalman filter for indoor localization," *Sensors*, vol. 15, no. 1, pp. 715–732, 2015.

[22] F. Pedregosa *et al.*, "Scikit-learn: Machine learning in Python," *J. Mach. Learn. Res.*, vol. 12, pp. 2825–2830, Oct. 2011.

[23] D. Guo, J. Chen, A. M. MacEachren, and K. Liao, "A visualization system for space-time and multivariate patterns (VIS-STAMP)," *IEEE Trans. Vis. Comput. Graphics*, vol. 12, no. 6, pp. 1461–1474, Nov. 2006.

[24] Z. Wu, C. Li, J. Ng, and K. Leung, "Location estimation via support vector regression," *IEEE Trans. Mobile Comput.*, vol. 6, no. 3, pp. 311–321, Mar. 2007.

[25] M. Rahman, Y. Park, and K. Kim, "RSS-based indoor localization algorithm for wireless sensor network using generalized regression neural network," *Arabian J. Sci. Eng.*, vol. 37, no. 4, pp. 1043–1053, 2012.

[26] D. P. Solomatine and D. L. Shrestha, "AdaBoost.RT: A boosting algorithm for regression problems," in *Proc. IEEE Int. Joint Conf Neural Netw.*, vol. 2. Jul. 2004, pp. 1163–1168.

[27] F. Vanheel, J. Verhaevert, E. Laermans, I. Moerman, and P. Demeester, "Automated linear regression tools improve rssi wsn localization in multipath indoor environment," *EURASIP J. Wireless Commun. Netw.*, vol. 2011, no. 1, p. 38, 2011.

[28] H. Chen, Y. Zhang, W. Li, X. Tao, and P. Zhang, "ConFi: Convolutional neural networks based indoor Wi-Fi localization using channel state information," *IEEE Access*, vol. 5, pp. 18066–18074, 2017.

YALI YUAN received the M.Sc. degree from the University of Lanzhou, Lanzhou, China, in 2015. She is currently pursuing the Ph.D. degree with the Telematics Group, Institute of Computer Science, University of Göttingen, Göttingen, Germany. Her research interests include various topics related to wireless networks, in particular, for the localization and security.

CHRISTIAN MELCHING is currently pursuing the B.Sc. degree in applied computer science with the Institute of Computer Science, University of Göttingen, Germany. He is currently a Student Assistant with the Technical Department, Institute of Psychology, University of Göttingen. His current research interests include machine learning and network localization techniques.

YACHAO YUAN received the M.Sc. degree from the University of Chongqing, Chongqing, China, in 2017. She is currently pursuing the Ph.D. degree with the Group of SMRG, Chair of Information Management, University of Göttingen, Göttingen, Germany. Her research interests include object detection and localization, and structures management systems.

DIETER HOGREFE received the degree from the Philips Exeter Academy, USA, in 1976, and the Ph.D. degree in computer science and mathematics from the University of Hannover, Germany, in 1985. From 1983 to 1986, he was with the SIEMENS Research Center, Munich, where he was involved in the area of the analysis of telecommunication systems. He was responsible for the protocol simulation and the analysis of the CCS No. 7. From 1996 to 2010, he was the Chairman of the Technical Committee Methods for Testing and Specification, European Telecommunication Standards Institute, ETSI. From 2011 to 2013, he was the Dean of the Faculty of Mathematics and Computer Science. He has been a Full Professor (C4) with the Telematics Group, University of Göttingen, since 2002. Since 2003, he has been the Director of the Institute of Computer Science. He held full professor positions at the Universities of Bern, Luebeck, and Göttingen, and visiting positions at the University of Dortmund, Technical University Budapest, UC Berkeley, and Hamilton University. He published numerous papers and two books on Internet technology, security of wireless sensor networks, analysis, simulation, and testing of formally specified communication systems. His research activities are directed toward computer networks and communication software engineering.