Pre- and Post-Processing Algorithms with Deep Learning Classifier for Wi-Fi Fingerprint-Based Indoor Positioning

Amir Haider, Yiqiao Wei, Shuzhi Liu and Seung-Hoon Hwang *

Division of Electronics and Electrical Engineering, Dongguk University-Seoul, Seoul 04620, Korea; amirhaider@dongguk.edu (A.H.); weiyiqiao@dongguk.edu (Y.W.); nasecduke@hotmail.com (S.L.)

* Correspondence: shwang@dongguk.edu; Tel.: +82-2-2260-3994

Received: 28 November 2018; Accepted: 4 February 2019; Published: 8 February 2019

Abstract: To accommodate the rapidly increasing demand for connected infrastructure, automation for industrial sites and building smart cities, the development of Internet of Things (IoT)-based solutions is considered one of the major trends in modern day industrial revolution. In particular, providing high precision indoor positioning services for such applications is a key challenge. Wi-Fi fingerprint-based indoor positioning systems have been adapted as promising candidates for such applications. The performance of such indoor positioning systems degrade drastically due to several impairments like noisy datasets, high variation in Wi-Fi signals over time, fading of Wi-Fi signals due to multipath propagation caused by hurdles, people walking in the area under consideration and the addition/removal of Wi-Fi access points (APs). In this paper, we propose data pre- and post-processing algorithms with deep learning classifiers for Wi-Fi fingerprint-based indoor positioning, in order to provide immunity against limitations in the database and the indoor environment. In addition, we investigate the performance of the proposed system through simulation as well as extensive experiments. The results demonstrate that the pre-processing algorithm can efficiently fill in the missing Wi-Fi received signal strength fingerprints in the database, resulting in a success rate of 88.96% in simulation and 86.61% in a real-time experiment. The post-processing algorithm can improve the results from 9.05–10.94% for the conducted experiments, providing the highest success rate of 95.94% with a precision of 4 m for Wi-Fi fingerprint-based indoor positioning.

Keywords: Wi-Fi fingerprint; indoor positioning; IoT; data pre- and post-processing; deep learning

1. Introduction

In the past decade, indoor location-based services (LBSs) have attracted lots of attention, driving the development of indoor positioning technologies to fulfil the technological requirements of modern day communications services. One of the key research issues is the development of a precise and reliable indoor positioning system to improve the inadequate performance of current positioning systems in indoor environment [1]. While global positioning systems (GPS) are unavailable in indoor environments due to the low penetration power of microwaves in indoor locations, several other candidates such as radio waves, acoustic signals, magnetic field, Wi-Fi, and other sensors information from mobile devices [2–4] can serve the purpose of determining the user location in indoor environments. Among the mentioned technologies, Wi-Fi signals-based indoor positioning is the most popular due to its wide deployment and the low cost of Wi-Fi networks [5]. Wi-Fi is considered more suitable in complex environments. Wi-Fi-based positioning technology can be classified into two types, time-space attribute methods and received signal strength (RSS)-based methods [6]. The latter is also known as fingerprinting and is one of the most popular indoor positioning technologies.
Unlike measurement-based methods, fingerprint-based methods can be easily implemented without additional hardware. Generally, the process of Wi-Fi fingerprint-based indoor positioning consists of two phases: offline and online. In the offline phase, the radio map or fingerprint database is constructed by measuring and recording the RSS of Wi-Fi signals from different access points (APs). In the online phase, the real-time RSS measurements from the user are matched and compared with the fingerprints in the database via classification algorithms to estimate the position of the user [7]. The fingerprint database can be constructed using an IoT device or smartphone. This database is then used to train the classifier to be able to predict user position in a real indoor environment with desired target locations. A high-performance positioning server is in charge of database storage and position prediction (classification task). Machine learning and deep learning-based classifiers can be employed to handle this classification task at the positioning server [8–11]. Figure 1 shows the working procedure for a Wi-Fi fingerprint-based indoor positioning system.

Figure 1. Illustration of Wi-Fi fingerprint-based indoor positioning system.

In the literature, several Wi-Fi fingerprint-based indoor positioning systems have been proposed which intend to improve specific location prediction and the overall system success rate. An efficient indoor positioning technique is presented in [12,13] and employs Wi-Fi fingerprinting along with trilateration. The trilateration technique employs knowledge of distance for three known APs at every target location while constructing the RSS fingerprint database. Similarly, a contour-based trilateration technique was presented in [14] where the RPs with same signal levels were combined to form a contour and the location was estimated. This approach highly depends on the position and orientation of the desired APs. A novel indoor positioning scheme which operates robustly by extracting the multipath propagation delay profile of Wi-Fi signals was presented in [15]. With this approach, changes in the indoor environment like multipath propagation can be efficiently compensated to improve system robustness. An experimental evaluation of location methods based on RSS measurements was presented in [16]. This work focused on VHF frequencies and outdoor positioning system and did not take Wi-Fi RSS fingerprint-based indoor positioning into consideration. To improve the fingerprinting database construction, an algorithm termed “Slide” is presented in [17], where the RSS fingerprint is recorded using a smartphone. This algorithm is limited only to linear positioning, which is not the case in most complex-structured indoor environments. As mentioned earlier, the time-varying nature of Wi-Fi signals and time consumed in the development of the RSS fingerprint database are major constraints in adopting Wi-Fi RSS measurements for indoor positioning. To address this issue, a gradient-based approach to stabilize the RSS gradient of Wi-Fi signals was proposed in [18], but an accuracy of 5.6 m was reported, which is not very practical when the indoor environment is considered. The RSS measurements are prone to environmental dynamics such as weather conditions, which can result in degradation of accuracy of distance estimation. Considering this issue, the rain attenuation effect was included in the pathloss model in order to estimate the impact of precipitation in the outdoor environment [19]. To address time consumption in the construction of the RSS fingerprint database, an autonomous crowdsourcing approach using multiple smartphones is described in [20].
However, autonomous behaviour for database collection is implemented by employing an existing trusted portable navigator in the target location, which is also immune to variations in the indoor environment, resulting in an estimation error greater than 5.7 m, close to that of the above mentioned gradient-based approach. Another approach to reduce time consumption in RSS fingerprint database construction is to analytically estimate the RSS distribution instead of manual data collection. To estimate the RSS distribution for APs, a probabilistic approach is described in [21], where a single- and double-peak Gaussian distribution is employed for the AP signals. The extensive efforts to improve the accuracy of RSS-based centroid localization algorithm in indoor environment were presented in [22]. However, the focus was on wireless sensor network nodes and the indoor area for the experiment was limited to a single room of $6 \times 7$ square meters. The analysis cannot be generally applied for large area. An extensive overview of the offline evaluation of smartphone-based indoor positioning techniques demonstrated at International Conference on Indoor Positioning and Indoor Navigation (IPIN) competition 2017 was presented in [23]. The authors concluded that the database for fingerprint algorithm would be recorded in a static manner or at very slow continuous motion such as 0.2 m/s. Additionally, the issue of training database density was addressed in [24], where the fingerprint-based approaches were preferred with a dense database.

Both machine learning and deep learning-based classifiers are popular for handling the matching and classification tasks. Recently, the deep learning gains more attention because of its high accuracy with huge amount of trained data [25]. Compared to the machine learning techniques, the deep learning attempts to learn high-level features from data in an incremental manner, which can eliminate the need of domain expertise and hard core feature extraction [26]. Regarding the training time, the deep learning algorithms take longer time in the offline phase due to the large number of parameters. However, the scenario is totally reversed in the online phase where the deep learning takes much less time to run [27]. That is, the deep learning outperforms in the case of large data size. Keeping the above-mentioned aspects in minds, we employ the deep learning-based classifier in this work.

Several classifiers have been introduced to indoor localization to handle the matching task, which gives rise to a complicated multi-classes classification problem. To address the challenge of developing an efficient classifier for the positioning server, [28] presents several machine learning approaches including kNN, a rules-based classifier (JRip), and random forest, employing them for Wi-Fi fingerprint-based indoor positioning systems. The results indicate that the Random Forest classifier offers the best performance. Random forest is one of the most powerful machine learning algorithms available today [29]. It can handle both classification and regression tasks and has been widely used in various applications such as Internet traffic interception, voice and image classification. However, it has been rarely utilized in indoor positioning. An on-device learning approach is depicted in [30], where a mobile application allows users to build and offline train their own RSS maps. A deep learning-based classifier for indoor position is presented in [31]; however, it employs channel state information (CSI) instead of the Wi-Fi RSS value, which is our focus in this work. To the best of our knowledge, the only reported work on deep learning-based Wi-Fi fingerprinting is done by [32]; however, database collection is not addressed and the existing UJIIndoorLoc dataset [33] is employed.

The key issues with the offline phase of such Wi-Fi fingerprint-based indoor positioning systems are the high variations in Wi-Fi signals over time, fading signals due to multipath propagation caused by hurdles, people walking in the area under consideration and addition/removal of Wi-Fi APs [34–36]. Additionally, the signal strength and resource allocation by the APs is dependent on the number of connected users. As most Wi-Fi-based positioning technologies use the existing infrastructure of the Wi-Fi network in the indoor environment, these problems cannot be avoided as they require control of the infrastructure. It is important to mention that the performance of the positioning server highly depends on the quality of the RSS fingerprint database used to train the server for position prediction [37]. Most real world data is composed of inaccurate, noisy, inconsistent, and missing data. The reasons for the existence of such data could be technological problems stemming from gadgets that gather data, a human mistake during data entry and much more. The collected data cannot be
used directly for training the server. Some specified machine learning and deep learning models for the server need information in a specified format, for example, the random forest algorithm does not support null values, and therefore, execution of the random forest algorithm null values has to be managed from the original raw dataset. To solve this problem, data pre-processing is performed. Data pre-processing is an important part of data science [38]. It includes the two concepts “data cleaning” and “feature engineering”. These two are compulsory for achieving better accuracy and performance in the machine learning and deep learning algorithms. Data pre-processing is necessary because of the presence of unformatted real-world data. Similarly, during the online phase for Wi-Fi fingerprint-based indoor positioning, there can be some errors in position estimation for users as the test data might be affected due to limitations in the indoor environment. To address this issue, another data processing scheme can be applied and this process is termed data post-processing. Data post-processing is performed on the result data. Post-processing allows for analysis of the previous result sets in a formal manner. Data pre- and post-processing algorithms are expected to improve the performance of Wi-Fi fingerprint-based indoor positioning systems [39]. A pre-processing approach that involves removing useless APs and their respective RSS from the fingerprint database was presented in [40]. This can help in reducing the computational overhead during position estimation and improves the system performance. A data pre-processing technique was presented in [41] to remove noisy data from the RSS fingerprint database by combining Wi-Fi and geographical information system (GIS). To accommodate the heterogeneity of RSS measuring devices, a data pre-processing algorithm was presented in [42], which can scale the RSS values from various devices to a uniform range. Unlike the abovementioned server-based pre-processing algorithms, a user-based data pre-processing library employing RSS filtering was presented for android-based devices in [43]. To the best of our knowledge, there is no data pre-processing algorithm developed for filling in missing values in the Wi-Fi RSS fingerprint database, which is expected to improve the system performance. Similarly, a brief survey on post-processing algorithms for machine learning and data mining was presented in [44]. However, to the best of our knowledge, there is no post-processing algorithm available for deep learning-based classifiers for Wi-Fi fingerprint-based indoor positioning.

In order to solve the above problems with Wi-Fi fingerprinting, much research has been conducted on algorithms for data pre-processing to remove noisy and missing data, and many techniques for improving the classification server have been developed. However, conventional techniques that have already been developed still have limitations in securing high accuracy, reliable success percentage and compensation for environmental changes that cause performance degradation. To the best of our knowledge, data post-processing has not been described for Wi-Fi fingerprint-based indoor positioning in the literature. There is a need for the development of an algorithm which can efficiently tackle the above-mentioned challenges in real-time using the existing infrastructure of the indoor environment.

In this work, we present a Wi-Fi fingerprint-based indoor positioning system using a deep learning classifier. The main contributions of this work are as follows:

- A data pre-processing algorithm to compensate for impairments in the collected RSS fingerprint database;
- A data post-processing algorithm to enhance the system performance by limiting the effects of the indoor environment on the experimental phase of the proposed system; and
- Investigation of the performance of a deep learning-based classifier at the server in charge of the RSS fingerprint database storage and position prediction.

The rest of the paper is organized as follows: The proposed system model is described in Section 2. The simulation and experimental results for the proposed system are discussed in Section 3. Finally, Section 4 presents the conclusion of our work and future directions.
2. Proposed System

In this section, an overview of the proposed system is first presented. The environment and setup for simulation and experiment along with RSS fingerprint database construction is described in Section 2.2. Description of the data pre-processing algorithm is presented in Section 2.3. The deep learning classifier and data post-processing algorithm are described in Section 2.4 and 2.5, respectively.

2.1. Overview of the Proposed System

In this work, we propose a Wi-Fi fingerprint-based indoor positioning system equipped with data pre- and post-processing algorithms. Figure 2 depicts the actual two-phase indoor positioning service delivered by the proposed system. The proposed system is a server-based and user-active indoor positioning system. A high-performance positioning server is in charge of database storage and position prediction (classification task) by employing deep learning.

![Figure 2. Proposed Wi-Fi fingerprint-based indoor positioning system equipped with data pre- and post-processing modules.](image)

The proposed system consists of two phases: an offline and an online phase. Firstly, in the offline phase, the database is constructed by mapping the RSS measurements at each reference point (RP) through a smartphone. The database in the offline phase of Figure 2 consists of two kinds of database. One is training and another is trial database. The training database is used in the offline phase for training the deep learning classifier. The trial database is used to perform the offline test simulation to evaluate the performance of the deep learning classifier. Both training and trial databases are pre-processed using the proposed pre-processing algorithm. The pre-processed databases are then used to train the deep learning classifier, which turns to be the trained classifier in the online phase. Secondly, in the online phase of Figure 2, five real-time measurements at unknown user positions are pre-processed to fill the missing values. The pre-processed measurements are then passed to the trained classifier to determine the user location. The trained classifier returns five decisions for the user’s position which are then passed to the data-post processing algorithm. The post-processing algorithm selects the most frequent decision for user position based on the “Majority-rule”. Finally, the decision on user location is passed to the user’s smartphone.

2.2. Environment and Setup

To implement the proposed system model, RSS fingerprint database collection and experimental verification of the simulation data are both performed on the 7th floor of the New Engineering Building at Dongguk University, Seoul, Korea. As shown in Figure 3, the 52 × 32 square meters target area is
divided into 74 RPs with an interval of 2 m between two consecutive RPs. The pre-installed anonymous
APs on the 7th floor are used in this work. In Figure 3, the location of the APs is unknown to the
positioning server. The main motive is to use the pre-installed infrastructure of Wi-Fi APs in the
building for indoor localization. The positioning server is a Dell Alienware model P31e (Alienware,
hardware subsidiary of Dell, Miami, FL, USA). A Samsung SM-E310K smartphone is used to measure
the RSS value from all available APs at each RP.

On the software side, construction of the RSS fingerprint database, data pre-processing, deep
learning classifier, data post-processing, and online experiment program are all performed in Python
and the tensorflow platform.

To collect the RSS fingerprint database for the above environment, a smartphone is used to
measure the RSS value from all available APs at each RP. Each measurement contains the RP label,
time, date, number of available APs, MAC address and respective RSS value for each AP at every RP
on the floor map. This measurement, termed the fingerprint, is then transmitted to the positioning
server. Five RSS fingerprints are collected in forward as well as in backward directions at each RP. The
sampling time for each RSS measurement is 5 s, which means 25 s for total five measurements. For
74 reference points, the total time consumed 31 min for each direction and 62 min for both directions.
For the training data, the measurements are taken in the morning, afternoon and evening for seven
days, which results in 22 h approximately. For the trial data, the measurements are taken for two days
in the same manner. Therefore, the time consumption is 6.5 h, approximately. All these fingerprints are
saved as training and trial files in .csv format in the positioning server, as the deep learning classifier
at the server is designed to use .csv format files as input. At each RP, several APs are present with

![Floor Map of 7th Floor
New Engineering Building
Dongguk University, Seoul](image-url)
unique MAC addresses. In the .csv file, the MAC addresses are added on the first row at the header of the file. If the MAC address is already present, then the RSS value is added to the corresponding column. If the MAC address is not present, the value is added to a new column. For 74 RPs, the number of columns can, therefore, be more or less than 256. The number 256 would be the maximum corresponding to the input size of the deep learning classifier. Therefore, the training and trial .csv files must contain 257 columns, where 256 columns are used to save the RSS fingerprint while 1 column is used to indicate the RP label. After generating the database, the number of the column in both training and trial .csv files are checked. If the number is less than 256, the remaining columns are filled with “0 s”. Otherwise, extra columns over 256 are discarded, starting from the AP with the minimum number of RSS fingerprints. This collected RSS fingerprint database is then passed to the data pre-processing section as shown in Figure 2.

2.3. Pre-Processing Algorithm

As the collected dataset contains RSS fingerprints recorded at 74 RPs on the floor map, 256 APs are present in the RSS fingerprint database. However, for each individual RP measurement, the number of APs is around 20 on average, so the remaining spaces are filled with “0 s” while the database is prepared. As discussed earlier, some algorithms like random forest cannot perform well with null values, so zero values should be removed. To accomplish this task, a data pre-processing algorithm is proposed in this work. The flow chart for the proposed data pre-processing algorithm is depicted in Figure 4.

The pre-processing algorithm is designed to fill in the missing values previously recorded as “0 s” in the RSS fingerprint database for each RP measurement with the average values for each AP at 74 RPs. It is important to mention here that the average is taken only for the non-zero RSS values for each AP at 74 RPs. The steps followed in the pre-processing algorithm are below:

1. **Step 1:** Calculate the average of all non-zero values in each column (representing one AP) of the Training RSS fingerprint database (Training.csv);
2. **Step 2:** Check the training data file for “0” values;
3. **Step 3:** Replace zero with the average value from step1 at the corresponding row and column in the Training RSS fingerprint database;
4. **Step 4:** Repeat step 1 and step 3 for the trial fingerprint data (Trial.csv). (Note: the average of the training RSS fingerprint database is replaced for “0” values in Trial.csv).

Figure 4. Flow graph for the proposed pre-processing algorithm.
• Step 1: Calculate the average of all non-zero values in each column (representing one AP) of the Training RSS fingerprint database (Training.csv);
• Step 2: Check the training data file for “0” values;
• Step 3: Replace zero with the average value from step1 at the corresponding row and column in the Training RSS fingerprint database; and
• Step 4: Repeat step 1 and step 3 for the trial fingerprint data (Trial.csv). (Note: the average of the training RSS fingerprint database is replaced for “0” values in Trial.csv).

The motive behind the pre-processing is to replace the missing values (“0 s”) in the database with a certain computed value which is relatively closer to the RSS fingerprints available in the database. It would be helpful for the deep learning classifier to operate with certain RSS values rather than zeros. Since the number of RSS fingerprints of 15,540 in the Training.csv is larger than 60 in the Trial.csv, the average value of the Training.csv replaces the missing values (“0s”) in the Trial.csv. In this manner, the missing values in the RSS fingerprint database are replaced with a computed average value and the database is ready to be used for training the deep learning classifier.

2.4. Deep Learning Classifier

In this work, we employed a deep learning-based classifier to predict user position. To deal with the variant and unpredictable Wi-Fi signals, the positioning is cast in a convolutional neural network (CNN) structure that is capable of learning reliable features from a large set of noisy samples. The database required for the simulation and experiments is collected from the real world on different days and times to mimic the actual environment. The proposed positioning system is summarized in Figure 2. For the offline CNN training, a four-layer deep neural network (DNN)-based coarse localizer is trained to extract reliable features from a massive noisy RSS fingerprint database. The deep learning classifier consists of three convolutional layers (CL) and one fully-connected (FC) layer. A SoftMax layer is used for multi-class classification. The architecture for the deep learning classifier is depicted in Figure 5. To reduce the training time, the number of neurons decreases as we go deeper in the network. The .csv file is converted to input image of size 16 × 16 by using tf.reshape in tensorflow [45]. First, the CNN learns reliable high-level features automatically from a large set of widely fluctuating RSS fingerprints and avoids hand-engineering. Second, this structure is capable of learning useful features directly from labelled data. Third, the CNN-based estimator is more robust as it predicts the position utilizing the extracted high level features. Moreover, the structure has an advantage in handling massive RSS fingerprints since the prediction at the positioning stage does not rely on any searches in sample space but requires only the forward evaluation of a trained feed-forward neural network. For the online positioning, the pre-processed real-time RSS measurements at target RPs are fed into the classifier to obtain a final position estimate. The parameters for the deep learning classifier are summarized in Table 1.

![Figure 5. Network architecture for the deep learning classifier.](image-url)
Table 1. Summary of parameters used in the deep learning classifier.

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>No. of layers</td>
<td>4</td>
</tr>
<tr>
<td>Type of layers</td>
<td>3-CL, 1-FC</td>
</tr>
<tr>
<td>Input image size</td>
<td>$16 \times 16 = 256$</td>
</tr>
<tr>
<td></td>
<td>Layer1: 8192</td>
</tr>
<tr>
<td></td>
<td>Layer2: 4096</td>
</tr>
<tr>
<td></td>
<td>Layer3: 2048</td>
</tr>
<tr>
<td></td>
<td>Layer4: 1024</td>
</tr>
<tr>
<td>No. of Neurons</td>
<td></td>
</tr>
<tr>
<td>Layer1:</td>
<td>8192</td>
</tr>
<tr>
<td>Layer2:</td>
<td>4096</td>
</tr>
<tr>
<td>Layer3:</td>
<td>2048</td>
</tr>
<tr>
<td>Layer4:</td>
<td>1024</td>
</tr>
<tr>
<td>Batch size</td>
<td>1000</td>
</tr>
<tr>
<td>Learning rate</td>
<td>0.001</td>
</tr>
<tr>
<td>Training Epoch</td>
<td>2000</td>
</tr>
<tr>
<td>Training probability</td>
<td>0.5</td>
</tr>
<tr>
<td>Inner-width</td>
<td>1024</td>
</tr>
<tr>
<td>Dropout rate</td>
<td>0.5</td>
</tr>
<tr>
<td>Location tolerance</td>
<td>2</td>
</tr>
<tr>
<td>Regularization ($\beta$)</td>
<td>0.01</td>
</tr>
<tr>
<td>nb_classes (SoftMax size)</td>
<td>74</td>
</tr>
</tbody>
</table>

2.5. Post-Processing Algorithm

After analysis of the output data, one other data processing scheme can be performed, which is termed data post-processing. The output data for the Wi-Fi fingerprint-based indoor positioning system needs to be post-processed as it might contain wrong decisions regarding the user location. For example, the output data for a Wi-Fi fingerprint-based indoor positioning system may contain the wrong target location as the environmental conditions or the signal strength of the APs may vary over time. This can cause the classifier of the Wi-Fi fingerprint-based indoor position system to select the wrong user location, hence reducing the success percentage and performance of the system. The post-processing algorithm can identify such variations in the output data of the Wi-Fi fingerprint-based indoor positioning system and compensate for imperfections in the indoor environment, such as variations in the signal strength level of Wi-Fi signals over time, lower levels of the RSS caused by multipath fading, changes in the Wi-Fi infrastructure due to addition or removal of Wi-Fi APs and hindrance created in the signal path due to people present in the vicinity of the user location. Post-processing eliminates wrong decision outputs from the indoor positioning system resulting from the above impairments in the indoor environment.

A data post-processing algorithm is proposed in this work to perform this task on the output data for the Wi-Fi fingerprint-based indoor positioning system. Figure 6 shows the steps used to implement the proposed post-processing algorithm. The proposed post-processing algorithm allows analysis of the classifier’s result sets in a formal manner. The algorithm employs “majority rule” to select the most frequent decision on the user’s position in the result data for the deep learning classifier. Following are the steps for the post-processing algorithm:

- **Step 1**: At every RP, five test RSS measurements are taken with the smartphone in forward as well as backward directions. The positioning server returns 5 predicted results for each RP.
- **Step 2**: The algorithm for post-processing computes the most frequent value among the five predictions by employing the “Majority rule”.
- **Step 3**: The most frequent value found during Step 2 is then declared as the final decision for the user location as RP number. The flow graph for the post processing algorithm is shown in Figure 6.
2.5. Post-Processing Algorithm

After analysis of the output data, one other data processing scheme can be performed, which is termed data post-processing. The output data for the Wi-Fi fingerprint-based indoor positioning system needs to be post-processed as it might contain wrong decisions regarding the user location. For example, the output data for a Wi-Fi fingerprint-based indoor positioning system may contain the wrong target location as the environmental conditions or the signal strength of the APs may vary over time. This can cause the classifier of the Wi-Fi fingerprint-based indoor positioning system to select the wrong user location, hence reducing the success percentage and performance of the system. The post-processing algorithm can identify such variations in the output data of the Wi-Fi fingerprint-based indoor positioning system and compensate for imperfections in the indoor environment, such as variations in the signal strength level of Wi-Fi signals over time, lower levels of the RSS caused by multipath fading, changes in the Wi-Fi infrastructure due to addition or removal of Wi-Fi APs and hindrance created in the signal path due to people present in the vicinity of the user location. Post-processing eliminates wrong decision outputs from the indoor positioning system resulting from the above impairments in the indoor environment.

3. Numerical Results

3.1. Simulation Results

The performance of the proposed system is quantified in terms of the training accuracy and success percentage. Training accuracy is the percentage of correct RP predictions without any margin of error when RSS database samples are reused as input to the trained classifier model [7]. The training accuracy for the positioning classifier is given as:

\[
\text{Training accuracy} = \frac{\text{No. of correct predictions}}{\text{Total no. of predictions}} \times 100 \quad (1)
\]

Success is the percentage of correct position prediction with a margin of error of 2 RPs when the trial database is used as input [46]. As there are 74 RPs in the database, therefore, the total number of samples is 74. The success count is incremented with every correct prediction. The success percentage for the classifier is given as:

\[
\text{Success percentage} = \frac{\text{Success count}}{\text{Total no. of samples}} \times 100 \quad (2)
\]

As mentioned earlier, the interval between two consecutive RPs is 2 m, so the precision of success is 4 m in this work. The simulation for the proposed Wi-Fi fingerprint-based indoor positioning system is carried out for the raw database as well as the pre-processed database.

To stress the need for data pre-processing, Figure 7a shows five RSS fingerprints for 20 APs taken at RP-1. It can be seen clearly that for several APs, there is missing data filled with “0 s” at the RSS fingerprint database construction stage. This results in a non-uniform RSS fingerprint at every RP. To solve this issue, the pre-processing algorithm is employed which can replace the “0 s” in the RSS fingerprint database with computed average values for each AP. Figure 7b shows the impact of data pre-processing on the RSS fingerprint database. It can be seen clearly that by filling in the missing values using the pre-processing algorithm, the RSS fingerprints at RP-1 become more uniform and, thus, the probability of error at the training stage of the classifier, as well as in simulation and experimental results can be significantly reduced. The simulation results for the raw database and
pre-processed database are presented in Table 2. With pre-processing, a gain of 1.5% can be achieved using the data pre-processing algorithm. The simulation results are depicted in Figure 8. The data points used to generate the graphs of Figures 7 and 8 are available at [47]. Concerning the computation overhead for pre-processing, it takes 6.5 s for the training and trial database to be pre-processed for the simulation. Even if such large overhead occurs due to the database size, the system performance is not affected since it only happens in the offline phase.

Figure 7. Five RSS fingerprints for 20 APs at RP-1 (a) raw database without pre-processing (b) database with pre-processing.

Table 2. Summary of simulation results.

<table>
<thead>
<tr>
<th>Simulation Type</th>
<th>Training Accuracy</th>
<th>Success Percentage</th>
</tr>
</thead>
<tbody>
<tr>
<td>Without Pre-processing</td>
<td>83.98</td>
<td>87.46</td>
</tr>
<tr>
<td>With Pre-processing</td>
<td>60.96</td>
<td>88.96</td>
</tr>
</tbody>
</table>
It is important to note that by employing the pre-processing algorithm, there is not only a gain in the success percentage of the system, but also the success percentage remains constant as the number of training epochs increase. Such consistency in success percentage is vital for the experiment as the trained classifier .meta file is used in the experiment. For the raw database, the success percentage decreases as the number of epochs increase. This degradation can be explained by looking at the training accuracy. As the training accuracy increases, the phenomenon of overfitting occurs. Overfitting happens when a model learns the detail and noise in the training database to the extent that it negatively impacts the performance of the model on trial data. It can be seen clearly in Figure 8 that the training accuracy of the pre-processed database is lower than that of the raw database. This helps to avoid overfitting of training data and, thus, results in a consistent success rate with an increasing number of training epochs.

3.2. Experiment Results

Since the pre-processed RSS fingerprint database shows better performance in the simulation, the trained model .meta file is saved for the epoch with the highest success percentage and used for the online real-time experiment. In the experiment, a smartphone carried by the user captures the real-time RSSs for available APs and sends them with MAC addresses to the positioning server. The server pre-processes the RSS fingerprint and employs the trained classifier model to predict the user position with the received RSS fingerprint. In the online phase, the computation overhead is negligible about 0.4 ms for each real-time RSS measurement. For five real-time RSS measurements, a total computation overhead of 2 ms is needed. To verify the simulation results, we conducted 10 experiments at different times and days to include the effect of environment change on the proposed system. In each experiment, the user carried a smartphone from RP-1 to RP-74, termed the “forward direction” and then from RP-74 to RP-1, termed the “backward direction”. At each RP, the RSS fingerprint measurement and position prediction are both performed five times continuously. The post-processing algorithm is applied to the five predictions received from the positioning server to determine the most frequent value by employing the “majority rule” to predict the user position. The post-processing introduces a negligible computation overhead of 0.03 ms with the majority rule. The detailed experiment results are presented in Table 3.
The experiment results obtained without employing the post-processing algorithm shows the highest success rate of 86.61%, which is 2.35% lower than the simulation results. One reason for this difference is that the changing of environments can result in different propagation paths for Wi-Fi signals and hence greatly affects the RSS fingerprints of APs at each RP. The real-time experiment was performed months after the RSS fingerprint database construction, and the test database used for simulation is split from the training database. To solve this issue, we employed the post-processing algorithm described in this work. The post-processing algorithm can mitigate the effects of environment changes by discarding incorrect decisions regarding user position and selecting only the most frequent decisions. As shown in Figure 9, the post-processing algorithm can improve the results from 9.05–10.94% for the conducted experiments. Furthermore, variations between multiple experiments are reduced from 5.69% to 4.73% by employing the post-processing algorithm. Other approaches to solve such environment variations will be included in future research.

<table>
<thead>
<tr>
<th>Experiment Index</th>
<th>Time</th>
<th>Direction</th>
<th>Success % Without Post-Processing</th>
<th>Success % With Post-Processing</th>
<th>Gain %</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Morning</td>
<td>Forward</td>
<td>88.64</td>
<td>97.29</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Backward</td>
<td>84.59</td>
<td>94.59</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Average</td>
<td>86.61</td>
<td>95.94</td>
<td>9.33</td>
</tr>
<tr>
<td>2</td>
<td>Afternoon</td>
<td>Forward</td>
<td>86.75</td>
<td>95.94</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Backward</td>
<td>83.78</td>
<td>94.54</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Average</td>
<td>85.26</td>
<td>95.24</td>
<td>9.98</td>
</tr>
<tr>
<td>3</td>
<td>Evening</td>
<td>Forward</td>
<td>84.05</td>
<td>94.59</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Backward</td>
<td>86.75</td>
<td>94.54</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Average</td>
<td>85.40</td>
<td>94.59</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>Morning</td>
<td>Forward</td>
<td>82.43</td>
<td>93.24</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Backward</td>
<td>84.05</td>
<td>94.59</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Average</td>
<td>83.24</td>
<td>93.91</td>
<td>10.67</td>
</tr>
<tr>
<td>5</td>
<td>Afternoon</td>
<td>Forward</td>
<td>84.83</td>
<td>94.59</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Backward</td>
<td>82.47</td>
<td>94.59</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Average</td>
<td>83.65</td>
<td>94.59</td>
<td>10.94</td>
</tr>
<tr>
<td>6</td>
<td>Evening</td>
<td>Forward</td>
<td>84.05</td>
<td>93.24</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Backward</td>
<td>85.68</td>
<td>94.59</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Average</td>
<td>84.86</td>
<td>93.91</td>
<td>9.05</td>
</tr>
<tr>
<td>7</td>
<td>Morning</td>
<td>Forward</td>
<td>82.61</td>
<td>91.89</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Backward</td>
<td>81.74</td>
<td>93.24</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Average</td>
<td>82.17</td>
<td>92.56</td>
<td>10.39</td>
</tr>
<tr>
<td>8</td>
<td>Afternoon</td>
<td>Forward</td>
<td>81.58</td>
<td>90.54</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Backward</td>
<td>82.33</td>
<td>91.89</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Average</td>
<td>81.95</td>
<td>91.21</td>
<td>9.26</td>
</tr>
<tr>
<td>9</td>
<td>Evening</td>
<td>Forward</td>
<td>80.54</td>
<td>90.54</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Backward</td>
<td>81.31</td>
<td>91.89</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Average</td>
<td>80.92</td>
<td>91.21</td>
<td>10.29</td>
</tr>
<tr>
<td>10</td>
<td>Morning</td>
<td>Forward</td>
<td>83.59</td>
<td>93.24</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Backward</td>
<td>84.55</td>
<td>94.59</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Average</td>
<td>84.07</td>
<td>93.91</td>
<td>9.84</td>
</tr>
</tbody>
</table>

Table 3. Summary of experimental results.
4. Conclusions

In this work, we propose a server-based user-active Wi-Fi fingerprint-based indoor positioning system using a deep learning classifier. The proposed system is equipped with data pre- and post-processing algorithms to reduce the degradation in performance caused due to limitations in the RSS fingerprint database and the indoor environment. Programs for fingerprint database construction using smartphones, data pre-processing, deep learning classifier, data post-processing, and online real-time experiment are developed. The proposed system is implemented and examined in a real-time indoor environment with 74 target locations. The simulation results show that the pre-processing algorithm can efficiently fill in missing RSS fingerprints in the database, resulting in a success rate of 88.96% with the error margin of 2 RPs. The proposed system equipped with a data post-processing algorithm is further verified with online real-time experiments. The proposed system is able to provide an indoor positioning precision of 4 m with a 95.94% success rate. Future work will focus on finding the reasons for the gap between the simulation and real-time experiment, and methods to increase the positioning performance of the proposed system.

Author Contributions: A.H., Y.W., and S.-H.H. contributed to the main idea of this research work. A.H. performed the simulations, experiments and database collection. S.L. performed experiments for the work. The research activity was planned and executed under the supervision of S.-H.H. A.H., and S.-H.H. contributed to the writing of this article.

Funding: This work was supported by the Technology Innovation Program (or Industrial Strategic Technology Development Program-Industrial site core technology Early development project) (10084909, Development of improved RTLS engine and IoT tag using machine learning technique reflecting real environments) funded by the Ministry of Trade, Industry, and Energy (MOTIE, Korea)

Acknowledgments: The authors would like to offer their sincere gratitude to Mr. Sang Moon Lee, CTO of JMP Systems, Korea, for providing the equipment for database collection and setting up the experimental environment.

Conflicts of Interest: The authors declare no conflicts of interest regarding the publication of this article.

References


© 2019 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (http://creativecommons.org/licenses/by/4.0/).