Fault-Tolerant indoor localization based on speed conscious recurrent neural network using Kullback–Leibler divergence

Pothuri Surendra Varma1 · Veena Anand1

Received: 5 August 2021 / Accepted: 5 February 2022 / Published online: 25 February 2022
© The Author(s), under exclusive licence to Springer Science+Business Media, LLC, part of Springer Nature 2022

Abstract
IoT services are the basic building blocks of smart cities, and some of such crucial services are provided by smart buildings. Most of the services like smart meters, indoor navigation, lighting control, etc., which contribute to smart buildings, need the locations of people or objects within the building. This gave rise to Indoor Localization, where only the infrastructure of the building has to be used for localization as accessing the Global Positioning System is difficult in indoor environments. Many approaches have been proposed to predict locations based on the infrastructure available indoors, and some of such techniques use Wi-Fi access points. Still, unfortunately, very few studies have concentrated on tolerating faults while being cost-effective. This work discusses hardware implementation of indoor localization. It then proposes a learning algorithm SRNN (Speed Conscious Recurrent Neural Network) that uses the RSSI (Received Signal Strength Indicator) values of available Wi-Fi access points in the building and predicts the location. Also, fault-tolerant approaches termed nearest RSSI and the most recent RSSI using Kullback–Leibler Divergence have been proposed to improve the location accuracy when access points go down and are prone to faults. Both the proposed approaches nearest RSSI and most recent RSSI along with SRNN improve the location accuracy by 4% and 2.1%, respectively, over existing techniques and contribute to optimizing predicted location's accuracy in Indoor Localization an IoT service for smart buildings.

Keywords Speed conscious recurrent neural network · Received signal strength indicator · Indoor localization · Smart buildings · Kullback–Leibler divergence

1 Introduction
Smart cities contribute to the quality of residents' lives by providing advanced services like intelligent transport, affective health monitoring, connecting people via information networks, waste collection management, promoting sustainable development, optimizing resources, intelligent parking slots, etc. Without having a specific target population, smart cities provide services to everyone, including children, older people, working women, sanitation staff, security officials, etc.

One of the major components that influence the growth of smart cities is smart buildings, as people spend most of their time indoors and the Internet of Things plays a crucial role in developing smart building applications. Smart Buildings can connect and automatically control various building operations through the Internet of Things like lighting control, regulating electrical usage, occupancy, efficient use of resources, intelligent security monitoring, location-based marketing services. They play a vital role in emergencies like dynamic routing to a safe exit in case of fire, smart navigation like locating the required store in a large shopping mall. Consider a situation where a five floored shopping mall contains around 100 stalls on each floor; no one can remember the locations of each stall and guide the visitors to the stall they want to go to, even if employees were recruited for such purposes, a machine easily outperforms them in such activities.

Location-based services are a primary asset for any smart building. Almost all of these mentioned operations require the location of people indoors, but accessing GPS (Global Positioning System) is difficult within indoor environments. Such difficulties gave rise to indoor localization...
techniques where the infrastructure of the building has to be used to provide localization. The fundamental challenge in such services is the accuracy of location.

The relationship between the RSSI (Received Signal Strength Indicator) and the device location is utilized as discussed in [1]. Different techniques, wireless technologies, architectures, and mechanisms used to achieve indoor localization were studied and described in [2, 3]. Figure 1 presents the basic architecture for indoor localization used in our proposal.

The mobile device which is connected to Wi-Fi provided by the smart building, reads the signal strength of each access point in the form of RSSI(Received Signal Strength Indicator) values; the RSSI values generally range from -100 to 0, the closer it is to 0, higher the signal strength. This list of RSSI values is sent to the server to predict the mobile device’s location.

The predicted location can be sent to the mobile device for navigation purposes and stored in a database for specific application usage and analysis purposes later on. The terms access points “or “anchor nodes” are used commonly to indicate the Wi-Fi devices that are fixed in the building from which the mobile device scans RSSI values. An RSSI fingerprint is defined as a vector of signal strengths from the visible anchor nodes to describe the radio signal characteristics at a specific location.

For predicting the location based on the list of RSSI values, a machine learning algorithm SCRNN is proposed, which gets trained when already collected data of RSSI values and locations are given to it and then can be used for testing where only RSSI values are given as input, and the locations are to be predicted. This approach has been discussed in Sect. 3.

One more challenge in indoor localization is that the mobile device may skip reading the RSSI values of one or more access points depending on its location. This may be due to temporary or permanent failure of a particular access point.

Figure 2 shows a scenario where RSSI values are received only from 3 access points, and the other one fails. Now only a list of 3 RSSI values is received by the server, and the challenge is it has to predict the location based on an algorithm that has already been trained with RSSI values of 4 access points. So definitely, location accuracy gets compromised because of such real-time issues. To handle such faults, ‘most recent RSSI’ and ‘nearest RSSI’ techniques are proposed, which are explained in Sect. 3.

Fingerprints are collected at a grid of known locations. These known locations are called reference points (RPs). Then the fingerprints and corresponding known location labels are stored in a database called a fingerprint database or radio map. The manual site survey is used to construct the fingerprint database, presented in Fig. 3.

A signal map (also known as a fingerprint database) is constructed by associating the collected fingerprints with physical locations.

The signal map that is already built is compared with the newly observed fingerprint for location determination. The approach to signal map construction—manual site survey—requires comprehensively exploring the whole area of interest and collecting fingerprints at a regular grid of survey points with known locations usually obtained with a physical floor plan [4].

1.1 Measurement techniques

1.1.1 Pedestrian dead reckoning

Pedestrian Dead Reckoning [5] is one of the solutions that can be used for indoor navigation. The term dead reckoning means the process of updating the value of any variable quantity by using an earlier value and adding whatever changes have occurred in the meantime. Analogous to this, the current position is computed using the previously determined position. Advancement of a position is done based on the known or estimated speeds over elapsed time and course.
Inertial measurement sensors like accelerometers, gyroscopes, and magnetometers in mobile devices provide directional acceleration information and are widely used for dead reckoning. Next position $P_n(x_n, y_n)$ from initial position $P_i(x_i, y_i)$ can be determined using (1) and (2).

$$x_n = x_i + \sum_{c=1}^{n} \alpha_c \cos \alpha_c$$

$$y_n = y_i + \sum_{c=1}^{n} \alpha_c \sin \alpha_c$$

$\alpha_c$ and $\lambda_c$ are the angle of orientation and step length obtained from inertial measurement units.

### 1.1.2 Wi-Fi access points

Wi-Fi technology determines user position based on the Received Signal Strength Indication information. A wireless local area network consisting of Wireless Access Points enables location computation in complex environments. RSSI collected from a group of access points at an instance forms a fingerprint. Distance ($D_{pq}$) between two fingerprints $R_p = [R_{p1}, R_{p2}, \ldots, R_{pj}]$ and $R_q = [R_{q1}, R_{q2}, \ldots, R_{qj}]$ is defined in (3).

$$D_{pq} = \sqrt{\sum_{i=1}^{j} |R_{pi} - R_{qi}|^2}$$

To address heterogeneity, adjustments can be made among two devices, P and Q using (4).

$$RSSI_Q = \alpha_1 RSSI_P + \alpha_2$$

$\alpha_1$ is close to 1, and $\alpha_2$ is the adjusted value. Therefore optimal RSSI offset ($O_{pq}$) during heterogeneity can be calculated through (5) and (6).

$$O_{pq} = \arg \min_{a_2} D_{pq}$$

$$D_{pq} = \sqrt{\sum_{i=1}^{j} |R_{pi} - R_{qi} - O_{pq}|^2}$$

### 1.1.3 Bluetooth beacons

Bluetooth Low Energy is a form of wireless communication explicitly designed for short-distance transmission. More recently, researchers started to use BLE based positioning method in indoor environments [6]. Indoor localization can be tracked by fusing the information from RSSI analysis from multiple BLE beacons.

The mobile device detects the signal from the beacon and can roughly calculate the distance from the beacon to estimate the location. The distance ($D$) can be computed using (7), where RSSI represents the signal strength received from the Bluetooth device.

$$D \approx 10^{\frac{(MP-RSSI)}{10}}$$

Fig. 3 Data set generation process through a manual site survey
where MP (Measured Power) is the industry calibrated read-
only value, which generally indicates the RSSI reading at
1 m distance. N is a constant within the range 2–4, which
depends on the environmental changes.

Still, there are more technologies like visible light infor-
mation, Radio Frequency Identification Devices, etc., that can be
used for estimating indoor location where as the three most
widely used technologies have been discussed. In this proposal,
Wi-Fi access points were used to perform localization.

2 Related work

Significant research has been done to optimize location
accuracy in indoor localization, and quite a few techniques
were proposed to handle faults. Some of such essential and
recent works were reviewed and mentioned as follows.

For Modelling RSS deviations more realistically, non-
Gaussian probability density functions were proposed in [7]
and Dempster-Shafer’s theory for combining and represent-
ing individual pieces of information provided by different
RSS sources regarding target node position. Prediction based
on CLTA (cooperative localization and tracking algorithm)
[8] is developed to improve position accuracy, [9] discusses
a weighted average filter method and using different types of
recurrent neural networks to enhance accuracy when tempo-
ral fluctuations of RSSI are encountered.

Mini-batch Singular Value Decomposition is used in [10]
to detect the target in case of sensor location uncertainty,
[11] has proposed multiple machine learning algorithms
to construct space of candidate labels, and user’s loca-
tion is estimated by label credibility. An array calibration
method [12] and Angle of Arrival estimation algorithm
were designed to achieve flexible orientation using limited
resources. TILoc (Torus Intersection Localization) [13] aims
to improve accuracy and robustness by mitigating noise
problems and positioning a target. A lightweight privacy-
preserving scheme (LWP²) [14] protects data privacy and
location privacy at a lower cost by minimizing the least
square error for an over-determined linear formulation.

A scalable Deep Neural Network Architecture [15]
with denoising autoencoder is proposed to achieve good
location performance, and high scalability, [6] uses two
methods –fingerprinting-based method and trilateration-
based method to accurately track human location in home
environments using Bluetooth Low Energy (BLE) devices.
CollabLoc [16] achieves cross-building localization and
privacy by collaborating signals from Wi-Fi and Cellular
RSSI, geomagnetic levels, light, and sound. SAWKNN
(Self Adaptive Weighted K Nearest Neighbors) technique
is proposed in [17] to achieve higher accuracy than normal
KNN, [18] uses Naive Bayes classifier and Channel State
Information to improve localization accuracy. Median
localization accuracy of 1 m is achieved in [19], where new
hardware is designed for an IoT device with a BLE interface,
accelerometer, and magnetometer sensors.

A fault-tolerant localization algorithm was developed in
[20] using K-means clustering and majority voting methods
[21] also proposed a fault-tolerant recurrent neural network
approach to improve location accuracy. Outstanding
improvements in location accuracy were achieved. This
is one of the latest and efficient researches in indoor
localization. But during momentary failure, when a base
station fails, location is predicted only based on remaining
active base stations by ignoring the failed base station.

Work [22] proposes a fault-tolerant area division stra-
 egy. The anchor deployment technique was used to generate
various shapes of sub-areas. The shipboard RSS signals
were investigated by conducting experiments in a real-world
shipboard environment, and good results were achieved [23].
focused on minimizing the number of required anchor nodes
for prediction, which works even during a node failure. This
is achieved using the Integer Linear Programming formu-
lation, which solved localization’s anchor node placement
problem based on trilateration. Better results were shown
optimizing the number of nodes, but results regarding loca-
tion accuracy were not presented [22, 23]. Focus more on
deploying the nodes at correct positions or minimizing the
required nodes rather than dealing with faults after they are
generated.

The Truncated Mean method pre-processes the data in
[24] to eliminate outliers on the valid data. The basic idea
is to eliminate the segments of minimum valued end and
maximum valued ends and then perform the mean of the
remaining data. After sorting the data collected in increasing
order, five percent of both minimum and maximum values
are removed.

A hierarchical framework using the K-means cluster-
ing algorithm and class-specific cost regulation extreme
machine classifier (CCR-ELM) [25] is proposed for large-
scale device-free localization. Another proposal, CRISloc
[26], uses Channel State Information as fingerprints and
(EEKNN) edge enhanced k nearest neighbors to automati-
 cally adjust the value of k and weights.

Also, some optimization techniques like EOMR [27], par-
ticle swarm optimization [28], and mathematical models of
[29–32] were studied to gain an understanding of how they
optimize in various networking scenarios and whether they
can be applied to maximize localization strategies.

A two-layered Indoor Localization System architecture by
considering the General Data Protection Regulation (GDPR)
as the grounding framework for preserving privacy through a
compliant Access Control (AC) system was proposed in [33]
to preserve social distance among people in indoor environ-
ments as prevention for covid-19 [34], also uses a two-phase
online localization process using MDT (Minimization of
Drive-Tests) data and a Bayesian classifier to reduce maintenance costs and manual collection.

The proposed method in [51] uses Channel State Information phases and amplitudes of various communication links for fingerprint construction to achieve Device-Free Localization (DFL). The probability of testing fingerprint belonging to each reference point is computed through a probabilistic approach that depends on Kullback–Leibler divergence. Then the resulting probabilities are used as weights for averaging the reference points for target localization. Least Squares Variance based Radio Tomography (LSVRT) is proposed in [52], which attempts to reduce the discrepancies generated by intrinsic motion, which may become noise to the localization model and increase the measured RSS variance. Instead of using offline calibration, a new online calibration technique is proposed to use online real-time measurements. The Kalman filter is used for tracking estimation on the proposed models.

Even though indoor localization can deliver timely information, the models proposed to perform localization still suffer from specific difficulties and pose challenges concerning fault tolerance, usability, and accuracy. The signal strengths get faded because of attenuation, and they are vulnerable to interference. Such noise reduces the accuracy of localization systems.

In terms of fault tolerance, there hasn’t been much research, and very few techniques were proposed to deal with faults. Moreover, the existing methods use additional hardware to improve location prediction accuracy during faulty conditions, which results in more installation and maintenance costs.

Alternate Deep Learning models were already proposed for the same problem scope but do not consider the speed consciousness, which surely assists in reducing localization error during faults. Alternate deep learning models include Convolutional Neural Networks and several variations of Recurrent Neural Networks like simple RNN, Long Short-Term Memory, and Gated Recurrent Unit [21]. A scalable Deep Neural Network Architecture achieves good location performance and high scalability by using autoencoder denoising [15]. Quite a few deep learning models already exist in indoor localization. However, our approach encourages further improvement in the scope of fault tolerance without additional costs.

Various notations used in the proposed model section are described in Table 1. A summary of some of the essential works is presented in Tables 2 and 3. Based on all this knowledge and also after reviewing a few more works [35–50], it was found that there has not been more research work that focused on detecting an invalid predicted location based on the speed of the pedestrians in an indoor environment. So, a model that depends on Recurrent Neural Network and uses the speed of pedestrian navigation to adjust the predicted location label is proposed, which is discussed in Sect. 3.

### Table 1: Description for various notations used

| Symbol | Description |
|--------|-------------|
| T      | Time ranging from 1 to t |
| RNN    | Recurrent Neural Network |
| R(t)   | RSSI given as input to RNN at time ‘t’ |
| PL(t)  | Previous Location at time t, used as the hidden state in RNN |
| F(t)   | Output of the Neural Network at time ‘t’ |
| M      | Loss function |
| W      | Weight matrices of the Neural Network |
| γ      | Learning rate to update the weight matrices |
| nbr(Ft) | Function to check if predicted location is present in neighbors of F, or not |
| b      | Number of neighbors returned from nbr() |
| D      | Data set |
| d      | Number of data points in the data set |
| x      | Total number of class labels |
| KLD    | Kullback–Leibler Divergence |
| NA     | RSSI value is not available |
| SRNN   | Speed Conscious Recurrent Neural Network |

3 Proposed model

#### 3.1 Localization based on speed conscious recurrent neural network

The proposed localization model constitutes of 3 phases (i) Generating the data set through manual site survey, (ii) Training phase where both vectors of RSSI values and actual locations are fed to the proposed algorithm, and (iii) The proposed SRNN predicts the locations when vectors containing only RSSI values are given.

The proposed Speed Conscious Recurrent Neural Network has inputs $R(T) = R(1), R(2), \ldots R(t)$, where $T$ represents time ranging from 1 to t. $R(t)$ is given as the input to RNN at the time ‘t’. $PL(T)$ gets used as a memory for the network (hidden state), representing the Previous Location in our implementation. The previously hidden form and present input are used to calculate the $PL(T)$.

$$PL(T) = f(U.R(T) + W.PL(T - 1))$$

(8)

$tanh$ non linear transformation is used as function $f$:

$$PL_t = tanh(W_pPL_{t-1} + W_RR_t)$$

(9)

Table 3 presents the parameters considered for optimization as mentioned in each research work.

The immediate last state in the recurrent neuron is considered; multiple such states can be used for longer
sequences. Output can be produced once the final state is determined.

\[ F_t = W_fPL_t \]  

For Loss Function M, gradients of weight matrices need to be updated using learning rate(γ).

\[ W = W - γ \frac{∂M}{∂W} \]  

Weights are updated so that the loss function is minimized.

The probability vector of outputs can be obtained using the softmax function in (12), where each neuron output is used as an argument.

\[ (Δf(t)M)i = \frac{∂M}{∂f_i(t)} = \frac{∂M}{∂f_i(t)} \frac{∂f_i(t)}{∂f_i(t)} \]  

RNN’s can use internal or hidden states to process the RSSI input sequences, as shown in Fig. 4. The speed consciousness is fed into the neural network by (13) that uses the current and previous output states to check the validity of prediction:

\[ \forall i \in \{1, ..., b\} if(F_{innbr(F(t-1))}) \rightarrow valid; else \rightarrow invalid \]  

(13)

If ‘invalid’ is returned, then \( F_t \) is updated to the neighbor of \( F_{t-1} \).

### 3.1.1 Kullback–Leibler divergence

Kullback–Leibler divergence provides relative entropy, which can identify the deviation of the faulty data with actual data and vice versa. Entropy can be used to measure the disorder of a data set which usually ranges between 0 and 1. Still, sometimes it can also be greater than 1, which

| Authors                  | Objectives                                      | Proposed methods                                      | Gap identified                                                |
|--------------------------|-------------------------------------------------|-------------------------------------------------------|--------------------------------------------------------------|
| Barsocchi et al. (2021)  | To preserve social distance in indoor environments | Privacy through a compliant Access Control (AC) system | Improvement in localization results to be compared with the state-of-the-art techniques |
| Achoeur et al. (2019)    | Location Accuracy, Computation time              | Non-Gaussian probability density function             | Noticeable costs due to specific hardware                     |
| Kumar and Das (2020)     | Target detection accuracy                        | Mini-batch Singular Value Decomposition               | Focus only on sensor location uncertainty                      |
| Li et al. (2020)         | Construct space of candidate labels              | Multiple Machine Learning Algorithms                  | A probabilistic model to estimate user’s location              |
| Sheng et al. (2020)      | Achieve flexible array orientation and receiver positions | Array calibration, AoA estimation algorithm           | Dynamic array arrangement might not be feasible in all environments |
| Li et al. (2020)         | Mitigate noise problem                           | Torus Intersection localization (TILoc)                | Not accurate during slight noise                               |
| Zhang et al. (2020)      | Lightweight privacy-preserving scheme (LPW²)     | Minimizing least square error for overdetermined linear formulation | More importance to privacy rather than location accuracy |
| Liu et al. (2020)        | Low cost, low power, and scalable localization   | DNN along with de-noising autoencoder                 | Performance is evaluated by analyzing only runtime and noise   |
| Bai et al. (2020)        | Low energy, accurate human location in home environments | Trilateration and fingerprint-based methods           | Additional hardware might be required as wearables are used   |
| Sadhu et al. (2019)      | Cross building localization, privacy, and latency | onion routing and perturbation/randomization techniques | Need to collaborate signals from Wi-Fi, Cellular RSSI, light, sound, geo-magnetic |
| Nieminen and Jrvinen (2020) | Privacy-preserving indoor localization scheme | Cryptographic primitives—Paillier encryption and garbled circuits | Costs of privacy increase noticeably |
| Zhao et al. (2020)       | Lightweight privacy-preserving localization      | Location Preservation Algorithm with Plausible Dummies | Attacks are still possible due to Spatio-temporal correlations among locations |
| Kwak et al. (2018)       | Low computational complexity, energy efficiency | BLE interface and two sensors—magnetometer and accelerometer | Issues using a magnetic field, costs of specific hardware |
| Bhat and Santhosh (2020) | Stable localization under faulty conditions      | K-means clustering and majority voting methods        | Conventional methods like K-means clustering were used         |
| Carvalho et al. (2019)   | Localization accuracy during faults              | Recurrent Neural Network                               | Failed base station is ignored completely during faults       |
indicates that there is a considerable amount of disorder, entropy can be defined as in (14):

$$\text{Entropy}(D) = \sum_{j=1}^{x} -q_j \log_2 q_j$$  \hspace{2cm} (14)$$

where $q_j$ represents the probability of occurrence of a class $j$ in the given data set and $x$ is the total number of classes. But this entropy can be used to compute the disorder of data sets individually and cannot relate one data set with another; Kullback–Leibler divergence (15) comes in handy for measuring the deviation in features before after faults.

$$\text{KLD}(A||B) = -\sum_{k=1}^{y} A(k) \ast \log \left( \frac{B(k)}{A(k)} \right)$$  \hspace{2cm} (15)$$

Operator $\ll$ represents A’s divergence from B, and $y$ is the total number of probability events. A and B are the probability distributions. Standard deviation can be computed by (16):

$$\sigma = \sqrt{\frac{1}{G} \sum_{i=1}^{G} (x_i - \mu)^2}$$  \hspace{2cm} (16)$$

where $G$ is the total size of the population, $\mu$ represents the mean and $x_i$ is each value of the data.

The advantage of using Kullback—Leibler divergence is unlike other divergences; it does not compute divergence symmetrically, i.e.,

$$\text{KLD}(A||B) \neq \text{KLD}(B||A)$$  \hspace{2cm} (17)$$

### Table 3 Table representing the parameters optimized in each work

| Authors                     | Location | Accuracy | Cost-effective | Energy efficient | Fault-tolerant | Privacy-preserving | Low latency |
|-----------------------------|----------|----------|----------------|------------------|----------------|--------------------|-------------|
| Fang et al. (2021) [34]     | ✓        | ✓        | X              | X                | X              | X                  | X           |
| Zhang et al. (2021) [25]    | ✓        | ✓        | X              | X                | X              | X                  | X           |
| Barsocchi et al. (2021) [33]| ✓        | X        | X              | X                | ✓              | X                  | ✓           |
| Achroufene et al. (2018) [7]| ✓        | X        | X              | X                | ✓              | X                  | ✓           |
| Luo et al. (2018) [8]       | ✓        | X        | X              | X                | X              | X                  | X           |
| Hoang et al. (2019) [9]     | ✓        | X        | X              | X                | X              | X                  | X           |
| Kumar and Das (2020) [10]   | ✓        | ✓        | X              | X                | X              | X                  | X           |
| Li et al. (2020) [11]       | ✓        | X        | X              | X                | X              | X                  | ✓           |
| Sheng et al. (2020) [12]    | ✓        | ✓        | X              | X                | X              | X                  | X           |
| Li et al. (2020) [13]       | ✓        | ✓        | X              | X                | X              | X                  | ✓           |
| Zhang et al. (2020) [14]    | ✓        | ✓        | X              | X                | ✓              | ✓                  | ✓           |
| Liu et al. (2020) [15]      | ✓        | ✓        | X              | X                | ✓              | ✓                  | ✓           |
| Bai et al. (2020) [6]       | ✓        | X        | ✓              | X                | X              | X                  | ✓           |
| Sathu et al. (2019) [16]    | ✓        | X        | ✓              | X                | X              | ✓                  | ✓           |
| Nieminen and Irvinen (2020) [37] | ✓        | X        | X              | X                | X              | ✓                  | X           |
| Zhao et al. (2020) [38]     | ✓        | ✓        | ✓              | ✓                | X              | X                  | X           |
| Kwak et al. (2018) [19]     | ✓        | X        | ✓              | ✓                | X              | X                  | X           |
| Bhat and Santosh (2020) [20]| ✓        | X        | X              | X                | ✓              | X                  | X           |
| Carvallo et al. (2019) [21] | ✓        | X        | X              | ✓                | X              | X                  | X           |
| Salazar et al. (2019) [39]  | ✓        | X        | ✓              | X                | X              | X                  | X           |
| Guidara et al. (2019) [40]  | ✓        | X        | X              | ✓                | X              | X                  | ✓           |
| Zhao et al. (2018) [41]     | ✓        | ✓        | X              | X                | ✓              | X                  | ✓           |
| Tiku and Pasricha (2019)     | ✓        | X        | X              | X                | X              | ✓                  | X           |

An unfolded Recurrent Neural Network was used for Localization.
3.1.2 Fault tolerance

Any access point can be prone to faults during the localization procedure. An efficient data pre-processing technique before passing data to the prediction algorithm can reduce the localization error up to a certain extent. Let the data set be represented as shown in (18).

\[
\begin{bmatrix}
R_{11} & R_{12} & \ldots & R_{1j} & C_k \\
R_{21} & R_{22} & \ldots & R_{2j} & C_k \\
\vdots & \vdots & \ddots & \vdots & \vdots \\
R_{i1} & R_{i2} & \ldots & R_{ij} & C_k
\end{bmatrix}
\]  

(18)

\[
\begin{bmatrix}
R_{11} & NA & R_{12} & NA & NA & R_{62} & NA & \ldots & R_{12}
\end{bmatrix}
\]  

(19)

NA indicates RSSI reading is not available. The proposed most recent RSSI fills the missing values by considering the RSSI value from its previous RSSI reading, as shown in (20).

\[
\forall i \in \{1, \ldots, d\} \quad \forall j \in \{1, \ldots, z\} \quad \begin{cases}
(\text{if } (R_{ij} = \text{NA})) & \text{then } R_{ij} \leftarrow R_{i-1j}
\end{cases}
\]  

(20)

For nearest RSSI, \(R_{i-1j}\) in (20) is replaced with \(R_{i\text{nr}}\), where \(R_{i\text{nr}}\) represents the RSSI value of the nearest access point within the same data point.

'd' is the total No. of data points, and 'z' is the total number of access points. Using the weighted average method, pre-processing the data set can be done using (21).

\[
\forall i \in \{1, \ldots, d\} \quad \forall j \in \{1, \ldots, z\} \quad \begin{cases}
(\text{if } (R_{ij} = \text{NA})) & \text{then } R_{ij} \leftarrow \frac{\sum_{k=1}^{z} w_{ik} R_{ik}}{z-1}
\end{cases}
\]  

(21)

The weights of each RSSI value can be computed using (22). If the pre-processing is done through simple average or mean of remaining access points without considering their weights, these Weights \((W_{ij})\) of (15) can be assigned to 1.

\[
\forall i \in \{1, \ldots, d\} \quad \forall j \in \{1, \ldots, z\} \quad W_{ij} = \frac{R_{ij}}{\sum_{k=1}^{z} R_{ik}}
\]  

(22)

3.1.3 Environment and dataset description

Experiments through hardware implementation were carried out in 34 feet long × 22 feet wide indoor environment, whose layout is shown in Fig. 5. Four mobile hotspots were used as access points (AP1, AP2, AP3, AP4), and another mobile device is used as the navigating device, scans the RSSI values from these 4 access points, and sends them to the server. An android application is developed to scan these RSSI values, which uses android.net.wifi.WifiManager class and sends them through URL to a computer having flask server implemented using python. The proposed algorithms were implemented in python, predicting location labels based on the RSSI values received.

The entire layout is divided into a grid of 16 locations, as shown in Fig. 5. Each row in the data set contains signal strengths (RSSI values ranging from -100 to 0) from 4 Wi-Fi access points and a location label (ranging from 1 to 16). 4000 such rows (data points) are present in the data set, out of which 3200 are used for training, and 800 are used for testing.

So the responsibility of the proposed approach is to predict one class label from 1 to 16 when readings from 4 access points are given as input. But during the testing phase, there might be faults. There is no guarantee that all 4 access points work as expected in real-time. So failures of each access point were also considered. While testing for fault tolerance, 400 data points from test data were supposed to be having such faults for experiment purposes.

To maintain speed consciousness in the proposed model, the nbr() function checks whether the current predicted location is a neighbor of the previous location or not. As it can be observed from Fig. 6, the distance between two neighboring locations is
locations, 4 and 9, is 5.4 feet, and that of non-neighboring nodes 4 and 10 is 10.8 feet. So the nbr() function uses this distance metric to check whether a person can travel from one location to the other within 1500 ms (the time interval between two data points). Obviously, at average walking speed, a person does not travel 10.8 feet/1500 ms. So this distance threshold is used to test whether the predicted location is valid or not compared to its previous location.

Apart from the generated data set, another data set from [35, 36] is also used for experimentation. According to [35], the environment is set up at an office location in Pittsburgh, USA. This office has seven Wi-Fi routers, and the signal strengths received from these routers categorize the user’s location in various office rooms.

An android device has been considered for capturing wireless signals, and these signal strengths were tabulated. The decision variable is one of the four rooms. It has 2000 data points which are now divided into 1500 training and 500 test data points. In this proposal, to test for fault tolerance, 400 data points out of 500 test points were made prone to faults where one RSSI value in each data point is randomly removed.

4 Results and discussion

The environment and the data set used to conduct experiments were already discussed in Sect. 3.1.3. The proposed algorithms were implemented in python using the scikit learn library. To compare the results of each approach, a data set is used that consists of 800 test data points in Sect. 4.1 and 500 test data points in Sect. 4.2. Each method uses each data point, predicts a location (or class label), and then compares whether this predicted location matches the actual location.

So finally, the number of locations "matched" and the number of locations "unmatched" out of total predicted locations are considered. An approach that generates less number of “unmatched” predicted locations and more number of “matched” predicted locations is said to be more efficient. The following terms will be used:

- Matched—predicted location is same as actual location.
- Unmatched—predicted location differs from the actual location (wrong or incorrect prediction).
- Location—one out of sixteen locations {1, 2, 3,…,16}.
- Datapoint number—each data point from the test data set.

The proposed algorithm is compared with machine learning algorithms used for localization like KNN (K-Nearest Neighbors), NB (Naïve Bayes), SV (Support Vector), and RNN (Recurrent Neural Network) algorithms.

4.1 Results using the generated data set through hardware implementation

Figure 7 presents a point graph of only unmatched predicted locations (in red-colored star-shaped marker) vs. actual room locations (in blue colored circle-shaped marker) of the test data for the proposed SRNN method, which resulted in 673 matched and 127 unmatched predictions, respectively.

It isn’t easy to study a graph if all the 800 test data points are presented, and in such cases, the markers used for predicted and actual locations appear to be at the same points for the human eye. It is almost impossible to find out where the unmatched locations are. So, only data points that generated unmatched predicted locations will be marked on the plot to make it easier to differentiate the actual and predicted location markers.

Figure 8 presents the number of unmatched locations individually at each location using SRNN. 0 (minimum) for location 12 and 22 data points (maximum) for location 9 were observed to be incorrect predictions.
Fig. 8 No. of wrong predictions at each location with SRNN

Fig. 9 No. of matched locations for each algorithm

Fig. 10 Plot for No. of matched locations during faults

Fig. 11 Comparison of results before and after faults

Fig. 12 Bar Graph presenting the improvement while using a fault-tolerant approach

Fig. 13 without faults vs. with faults vs. fault-tolerant approach
Figure 9 presents the results of SRNN compared to various algorithms, and the proposed SRNN algorithm has an improvement of 13 data points over the RNN method.

The grey colored lines in Fig. 10 indicate the same lines of Fig. 9, which are now compared to matched data points during faults (colored lines in Fig. 10).

Figures 10 and 11 show how the matched locations get reduced when nodes become faulty. The prediction becomes less efficient by around 37% (average for all the algorithms).

The nearest RSSI fault-tolerant technique fills the missing values before passing data to the algorithm. The improvement in prediction is shown in Fig. 12. The Number of matched locations improved by 28% (average for all the algorithms) compared to the data with faults.

The overall change in the number of correct predictions before and after faults and when using the proposed fault-tolerant approach can be observed in Fig. 13.

It can be observed from Fig. 13 how the number of wrong predictions changes before and after faults. Initially, there are 127 wrong predictions; faults increase to 376. However, they decrease to 267 when the fault-tolerant approach is used.

As presented in Fig. 14, the proposed fault-tolerant approaches nearest RSSI and most recent RSSI and SRNN improve the location accuracy by 4% and 2.1%, respectively.

Kullback–Leibler Divergence as in Fig. 15 becomes more (6.06) when the data is prone to faults; later, when the data is pre-processed, the divergence reduces to 3.29. Location prediction improves with the reduction in divergence.

It can be observed from Table 4 that Standard Deviation does not show much deviation between original data (11.27) and fault-tolerant data (11.70). However, KLD shows a divergence of 3.29 units, which means there can still be differences between original and fault-tolerant data predictions.
Results were also compared with KLDPA (Kullback–Leibler Divergence-based Probabilistic Approach) and LSVT (Least Squares Variance based Technique), presented in Figs. 16 and 17, respectively. KLDPA produces 608 matched locations, and LSVT generates 629 matched locations which are less efficient when compared to RNN and SRNN, as shown in Fig. 18.

4.2 Results using the available data set [35, 36]:

Graphs in this section are results of experiments conducted through the already available data set [35, 36], which has RSSI readings of 7 devices.

Figure 19 shows how the matched locations get reduced when a node becomes faulty. The prediction becomes less efficient by around 27% (average for all the algorithms).

The results of already existing fault-tolerant techniques along with the proposed schemes are presented in Figs. 20 and 21, which includes the Weighted Average method and Truncated Mean method. Truncated Mean outperforms the Weighted Average method, and the proposed nearest RSSI method and the Speed Conscious Recurrent Neural Network predict 5% better than the Truncated Mean method.

| Table 4: Divergence values for various data used |
|------------------------------------------------|
| Data     | Standard Deviation | Data        | Kullback–Leibler Divergence |
|----------|--------------------|-------------|-----------------------------|
| Original | 11.27              | Original vs. Faulty | 6.06                        |
| Faulty   | 14.87              | Fault-Tolerant vs. Faulty | 4.88                        |
| Fault-Tolerant | 11.70           | Original vs. Fault Tolerant | 3.29                        |

Fig. 16 Kullback–Leibler Divergences at various stages of data preprocessing

Fig. 17 Unmatched locations using KLDPA

Least Squares Variance based Technique

Fig. 18 Unmatched locations using LSVT

Fig. 19 Unmatched locations using LSVT
The improvement in results while using SRNN with fault tolerance can be observed in Fig. 22. In terms of running time, SRNN has taken approximately 740 ms more time than RNN for predicting 500 data points which can be negligible.

5 Conclusion

This paper describes the real-time implementation of indoor localization using minimum hardware possible, then proposes a Speed Conscious Recurrent Neural Network that monitors pedestrians’ navigation speed to adjust the predicted location. Experiments reveal that the proposed SRNN algorithm has improved location accuracy over the conventional Recurrent Neural Network technique.

The most recent RSSI method and nearest RSSI proposed for tolerating faults work by filling missing values intelligently, making the proposal cost-effective, unlike the traditional fault tolerating methods that use specific hardware. The most recent RSSI technique optimizes the location accuracy by 2.1%, and it can be implemented when temporary fluctuations occur for multiple access points. Whereas the nearest RSSI technique optimizes the location accuracy by 4% and can be implemented in situations where one access point fails continuously. The experiments were conducted in python using scikit learn and Keras libraries with Spyder tool; flask server is used to communicate with navigating mobile and the server.

There is still scope for further research that can concentrate on the deployment of access points and automatic adjustment of access point positions through mathematical models to ensure more signal coverage. Considering the problem scope and the research applied here, focus can also be made on regulating the frequency of RSSI scans and server communication to improve energy efficiency. More research opportunities regarding privacy and security emerge with the wide use of localization to ensure user location is not disclosed to unauthorized persons.

Supplementary information  The online version contains supplementary material available at https://doi.org/10.1007/s12083-022-01301-y.

Declarations

Conflict of interest  On behalf of all authors, the corresponding author states that there is no conflict of interest.
References

1. Gu Y, Ren F (2015) Energy-Efficient Indoor Localization of Smart Handheld Devices Using Bluetooth. IEEE Access 3:1450–1461. https://doi.org/10.1109/ACCESS.2015.2441694
2. Zafari F, Gkelias A, Leung KK (2019) A survey of indoor localization systems and technologies. in IEEE Communications Surveys & Tutorials 21(3):2568–2599 thirdquarter. https://doi.org/10.1109/COMST.2019.2911558
3. Sadowski S, Spachos P (2018) RSSI-Based Indoor Localization With the Internet of Things. IEEE Access 6:30149–30161. https://doi.org/10.1109/ACCESS.2018.2843325
4. Liang Q, Liu M (2020) An Automatic Site Survey Approach for Indoor Localization Using a Smartphone. IEEE Trans Autom Sci Eng 17(1):191–206. https://doi.org/10.1109/TAES.2019.2918030
5. Pan M-S, Li K-Y (2021) ezNavi: An Easy-to-Operate Indoor Navigation System Based on Pedestrian Dead Reckoning and Crowdsourced User Trajectories, in IEEE Trans Mob Comput 20(2):488–501. 1. https://doi.org/10.1109/TMC.2019.2946821
6. Bai L, Citaregga F, Bond R, Mulvenna M (2020) A Low Cost Indoor Positioning System Using Bluetooth Low Energy. IEEE Access 8:136858–136871. https://doi.org/10.1109/ACCESS.2020.3012342
7. Achroufene A, Amitay I, Chibani A (2018) RSS-Based Indoor Localization Using Belief Function Theory. IEEE Trans Autom Sci Eng 15(3):1163–1180. https://doi.org/10.1109/TASE.2018.2873800
8. Luo J, Zhang Z, Liu C, Luo H (2018) Reliable and Cooperative Target Tracking Based on WSN and WiFi in Indoor Wireless Networks. IEEE Access 6:24846–24855. https://doi.org/10.1109/ACCESS.2018.2830762
9. Hoang MT, Yuen B, Dong X, Lu T, Westendorp R, Reddy K (2019) Recurrent Neural Networks for Accurate RSSI Indoor Localization. IEEE Internet Things J 6(6):10639–10651. https://doi.org/10.1109/JIOT.2019.2940368
10. Kumar S, Das SK (2020) Target Detection and Localization Methods Using Compartmental Model for Internet of Things. IEEE Trans Mob Comput 19(9):2234–2249. https://doi.org/10.1109/TMC.2019.2921537
11. Li L, Guo X, Ansari N (2020) SmartLoc: Smart Wireless Indoor Localization Empowered by Machine Learning. IEEE Trans Ind Electron 67(8):6883–6893. https://doi.org/10.1109/TIE.2019.2931261.
12. Sheng M, Zheng L, JY, Valace S, Li J (2020) Accurate Indoor Localization Assisted With Optimizing Array Orientations and Receiver Positions. in IEEE Trans Veh Technol 69(1):509–521. https://doi.org/10.1109/TVT.2019.2951022.
13. Li H, Qian Z, Tian C, Wang X (2020) TiLoc: Improving the Robustness and Accuracy for Fingerprint-Based Indoor Localization. in IEEE Internet of Things 7(4):3053–3066. https://doi.org/10.1109/JIOT.2020.2964875.
14. Zhang G, Zhang A, Zhao P, Sun J (2020) Lightweight Privacy-Preserving Scheme in Wi-Fi Fingerprint-Based Indoor Localization. in IEEE Systems 14(3):4638–4647. https://doi.org/10.1109/JSYST.2020.2979790.
15. Liu C, Wang C, Luo J (2020) Large-Scale Deep Learning Framework on FPGA for Fingerprint-Based Indoor Localization. IEEE Access 8:65669–65617. https://doi.org/10.1109/ACCESS.2020.2985162
16. Sadhu V, Zouonou S, Sritapan V, Pompili D (2019) CollaLoc: Privacy-preserving Multi-modal Collaborative Mobile Phone Localization. in IEEE Trans Mobile Comput. https://doi.org/10.1109/ TMC.2019.2937775.
17. Hu J, Liu D, Yan Z, Liu H (2019) Experimental Analysis on Weight $S[K]$ -Nearest Neighbor Indoor Fingerprint Positioning. IEEE Internet Things J 6(1):891–897. https://doi.org/10.1109/JIOT.2018.2864607
18. Wu Z, Xu Q, Li J, Fu C, Xuan Q, Xiang Y (2018) Passive Indoor Localization Based on CSI and Naive Bayes Classification. IEEE Transactions on Systems, Man, and Cybernetics: Systems 48(9):1566–1577. https://doi.org/10.1109/TSMC.2017.2679725
19. Kwak M, Park Y, Kim J, Han J, Kwon T (2018) An Energy-efficient and Lightweight Indoor Localization System for Internet-of-Things (IoT) Environments. Proc ACM Interact Mob Wearable Ubiquitous Technol 2(1):17–28. https://doi.org/10.1145/3191749
20. Bhut SJ, Santhosh KV (2020) Fault Tolerant Localization Based on K-means Clustering in Wireless Sensor Networks. IEEE International Conference on Electronics, Computing and Communication Technologies (CONCECT), Bangalore India pp 1–5. https://doi.org/10.1109/CONCECT.2020.9198415.
21. Carvalho EC et al (2019) Towards a Smart Fault Tolerant Indoor Localization System Through Recurrent Neural Networks. International Joint Conference on Neural Networks (IJCNN), Budapest, Hungary, pp 1–7. https://doi.org/10.1109/IJCNN.2019.8852007.
22. Liu K, Mozi Chen E, Cai JM, Liu S (2018) Indoor localization strategy based on fault-tolerant area division for shipboard surveillance. Autom Constr 95(2018):206–218. https://doi.org/10.1016/j.autcon.2018.08.009.
23. Balc K, Prevostini M, Malek M (2015) Optimizing Sensor Nodes Placement for Fault-Tolerant Trilateration-Based Localization. IEEE 21st Pacific Rim International Symposium on Dependable Computing (PRDC), Zhangjiagai, pp 109–118. https://doi.org/10. 1109/PRDC.2015.23.
24. Yang Q, Zheng S, Liu M et al (2019) Research on Wi-Fi indoor positioning in a smart exhibition hall based on received signal strength indication. J Wireless Com Network 2019:275. https://doi.org/10.1109/13638-019-1601-3
25. Zhang J, Xiao W, Li Y (2021) Data and Knowledge Twin Driven Integration for Large-Scale Device-Free Localization, in IEEE Internet of Things J 8(1):320–331. https://doi.org/10.1109/JIOT. 2020.3005939.
26. Gao Z, Gao Y, Wang S, Li D, Xu Y (2021) CRISLoc: Reconstructable CSI Fingerprinting for Indoor Smartphone Localization, in IEEE Internet of Things J 8(5):3422–3437. https://doi.org/10. 1109/JIOT.2020.3022573.
27. Jaiswal K, Anand V (2020) EOMR: An Energy-Efficient Multi-path Routing Protocol to Improve QoS in Wireless Sensor Network for IoT Applications. Wireless Pers Commun 111:2493–2515. https://doi.org/10.1007/s11277-019-07006-x
28. Anand V, Pandey S (2017) Particle swarm optimization and harmony search based clustering and routing in wireless sensor networks. Int J of Computer Intelligence Syst 10.1252. https://doi. org/10.2991/ijcis.10.1.8.4
29. Chandrakar P (2019) A secure remote user authentication protocol for healthcare monitoring using wireless medical sensor networks. Int J Amb Comput Intelligence 10.96-116. https://doi.org/10.4018/IJACI.2019010106
30. Chandrakar P, Jain A, Balivada S, Ali R (2019) A Secure Authentication Protocol for Vehicular Ad-Hoc Networks. IEEE In Conference on Electric, Comput Commun Technol (ICCECT), Coimbatore India pp 1–7. https://doi.org/10.1109/ICCECT.2019.8869101.
31. Naik KJ, Naik DH (2020) Minimizing Deadline Misses and Total Run-time with Load Balancing for a Connected Car Systems in Fog Computing. Int J Scalable Comput Pract Experience, ISSN 1895–1767, ©2020 SCPE. 21(1)73–84. https://doi.org/10.12694/ scpe.v21i1.1616.
32. Naik KJ (2018) Workflow Scheduling Algorithm in Multi-Tenant Cloud Computing Environments (NCWSA). Int J Pure Appl Math 120(6):3563–3568.
33. Barsocchi P, Calabrò A, Crivello A, Daoudagh S, Furfari F, Girolami M, Marchetti E (2021) “COVID-19 & privacy: Enhancing of indoor localization architectures towards effective social distancing 9. https://doi.org/10.1016/j.array.2020.100051.
