Article

Sensor-fusion for Smartphone Location Tracking using Hybrid Multimodal Deep Neural Networks

Xijia Wei [1], Zhiqiang Wei [1] and Valentin Radu [2]

[1] University of Edinburgh;
[2] University of Sheffield;

Abstract: Many engineered approaches have been proposed over the years for solving the hard problem of performing indoor localisation using smartphone sensors. However, specialising these solutions for difficult edge cases remains challenging. Here we propose an end-to-end hybrid multimodal deep neural network localisation system, MM-Loc, relying on zero hand-engineered features, learning them automatically from data instead. This is achieved by using modality-specific neural networks to extract preliminary features from each sensing modality, which are then combined by cross-modality neural structures. We show that our choice of modality-specific neural architectures is capable of estimating the location with good accuracy independently. But for better accuracy, a multimodal neural network fusing the features of early modality-specific representations is a better proposition. Our proposed MM-Loc solution is tested on cross-modality samples characterised by different sampling rates and data representation (inertial sensors, magnetic and WiFi signals), outperforming traditional approaches for location estimation. MM-Loc elegantly trains directly from data unlike conventional indoor positioning systems, which rely on human intuition.

Keywords: Indoor Localization, Sensor Fusion, Multimodal Deep Neural Network, Multimodal Sensing, WiFi Fingerprinting, Pedestrian Dead Reckoning

1. Introduction

With a growing number of mobile applications requiring contextual information to adapt their services to user needs, location estimation is becoming an important feature of those tasks. GPS is a system-level navigation method relying on satellite signals. However, indoor environments are often sheltered from satellite signals. As a result, scientists have proposed alternative methods for indoor positioning relying on signals such as WiFi, Bluetooth, Infrared RSS and inertial movement sensors (accelerometer, gyroscope, barometer) [1]. Most of these solutions have been hard-engineering work, but this is becoming harder to adapt for edge cases of performance when the indoor environment changes.

There are two main approaches for performing indoor localization that have been at the core of most proposed systems: Pedestrian Dead Reckoning (PDR) and WiFi based Location Estimation (with the popular version known as WiFi Fingerprinting) [2]. In PDR, the starting location is assumed to be known and using specially engineered techniques (e.g. well-defined sets of formulations) to estimate the travelled distance and the direction of movement for predicting the following position. WiFi Fingerprinting systems compare the received signal strength with pre-recorded WiFi radio maps to estimate the most similar location.

However, these specially engineered systems fail to work when facing imperfect inertial sensor data (accelerometer, gyroscope and magnetometer). The drift and noise from sensors result in variations in the signal distribution, leading to lower accuracy in estimating the location. This limitation has encouraged different creative engineering solutions, such as particle filters [2,3], Kalman filters, graph-conditions [4] and constraint

Citation: Xijia, Wei.; Zhiqiang, Wei.; Valentin, Radu. Title. *Sensors* **2021**, *1*, 0. https://doi.org/

Received:
Accepted:
Published:

modelling [5]. These systems rely on mathematical formulations to make firm assumptions about the possible action conditions based on isolated observations in the sensor state. To enable these services in complex environments, systems are tuned to each deployment environment [4,6]. Nevertheless, these hand-engineered mathematical models lose efficiency when the indoor environment changes potentially with modified radio propagation distribution. Currently, no one system has reached the level of acceptance as GPS has for outdoor environments. This is due to the uncertainty about the indoor environments. It is costly to accommodate all possible actions with manual effort for system recalibration [7].

Many other approaches aim to combine the two approaches with ingenious solutions for fusing the sensors and independent estimations. These environment-oriented engineered positioning systems make firm assumptions about the possible conditions on isolated observations in data, many times restricting the possible users' activities because these may be harder to model with current mathematical formulations, or irregular movement samples would even disprove the initial models. Otherwise, these hand-engineered mathematical models fail to work when facing unpredictable situations.

Instead of the aforementioned tediously engineered algorithms, we aim to achieve a more convenient approach by relying on an end-to-end multimodal deep neural network (MDNN). We believe that moving the focus from minutely understanding mobility patterns to learning cross-sensor patterns automatically from data is a more attractive proposition. This will allow positioning systems to be more flexible and robust by continuous updating the models in training with fresh data.

Although multimodal machine learning has approved its success in modality-fusion tasks such as audio-vision speech recognition and contextual recognition [8], investigating multimodal machine learning on multisensory data in the field of location tracking for GPS-denied environments is still under exploration. To the best of our knowledge, this is the first proposed system using hybrid multimodal deep neural networks to perform a half-way feature fusion of cross-sensing modalities available on modern mobile phones (WiFi and inertial sensors).

The contributions of this work are as follows:

- We replace the traditional methods of operating on sensor data for location estimation with end-to-end machine learning approaches. This avoids hand-tuning the models to approximate the data. Instead, data representations are learned automatically from data.
- We deploy a recurrent neural network of Long short-term memory (LSTM) to model the sequential chain of estimations, which performs similarly to PDR – estimating a sequence of locations, starting from a known point and estimating the following points in the sequence from sensor data.
- Introducing the first use of modalities fusion with different neural network constructions on each sensing modality to extract intra-modality features, which are then combined by higher layer features across modalities.

2. Motivation and Related Work

Position estimation of smartphones inside buildings is a challenging task due to the GPS being unreliable in environments shielded by walls and ceilings. At the same time, other radio signals with longer penetration (cellular and FM) are limited to the granularity of position estimation they can offer [9]. Alternative methods have been proposed to take advantage of a broader range of sensors available on smartphones [10]. However, none have managed to produce a robust and scalable system for efficient indoor position estimation. We believe the reasons are: i) Indoor spaces are too complex to model with limited and fragmented observations from the environment (limited data), ii) Signal distortion occurs during propagation across the spectrum, from light and sound to radio frequency and magnetic, and iii) current systems rely on human interpreted

features extracted from data (e.g., engineered solutions to estimate the number of steps and direction of movement).

Machine learning has been given as a promising option due to its validated performance in several fields, including computer vision, natural language processing and pattern recognition that make inferences with noisy data [11]. It shows the advantages of learning the correlation between hidden features from the hyper-dimensional sensory data and target labels automatically [12].

Using machine learning for indoor location tracking has become a popular research topic of predicting target locations based on sensed data. However, many of them are investigated on a single modality, which limits their applications. Given the uncertainty of the signal distribution in indoor environments, model signal propagation from scarce observations is relatively challenged with simplistic modelling techniques. Meanwhile, the indoor environment is formed by multiple modalities, mono-modality limits the representation of indoor situations, which makes single-modality based machine learning positioning systems fail to work under extreme edge cases.

In order to enhance the capability of the artificial intelligence to understand the location tracking task from different perspectives, we propose a multimodal machine learning approach that allows neural networks to capture in-depth features of natural representations and learn from correspondences within sensor fusion data that merging inertial sensor data and RSS data as uniform data input for system inference with flexibility and robustness.

We believe that by relying on models with high generalisation to learn directly from data, taking advantage of growing data volumes, deep neural networks can tackle the aforementioned long-standing problems that limiting indoor localization by a purely data-driven approach.

Therefore, our purpose is to convert from the engineering problem to an artificial intelligence approach that shifting focus from identifying patterns manually and mathematically fitting propagation patterns to delegate computers with multimodal machine learning.

2.1. Pedestrian Dead Reckoning on Inertial Sensors

PDR builds on inertial sensors to estimate displacement distance and direction of movement. However, these sensors have their limitations. Sensor drift is one of the most notorious problems, making it hard to double integrate acceleration to estimate displacement [13]. The same problem is experienced when estimating the orientation of the movement. Figure 1 shows the drift in gyroscope readings for estimations of orientation on the movement in a straight line. We observe that in just a few seconds, the accumulating sensor drift is substantial, changing the estimated direction of the movement. Hence, relying on the gyroscope alone is known to be inefficient, so engineered solutions use extra sensor inputs for recalibration [14,15].

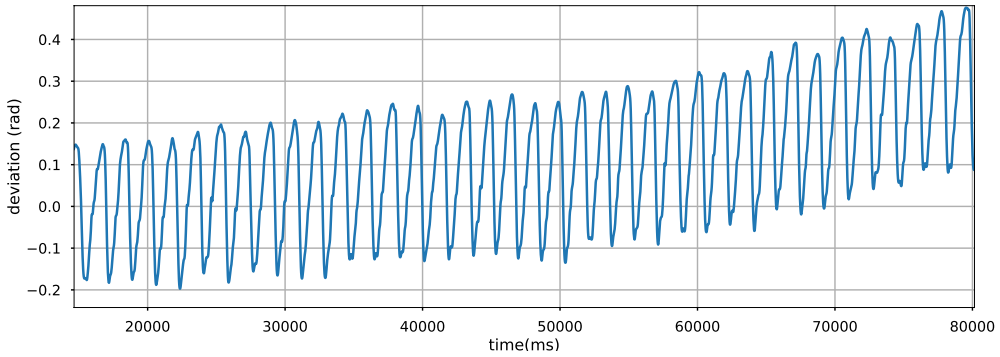


Figure 1. Gyroscope drifts disturbing direction calculation when sampling from a straight-line walking.

To replace hard and rigid engineered solutions, others use machine learning to identify sampling characteristics in inertial sensors, such as for step size estimation [16].

2.2. WiFi Fingerprinting on Received Signal Strength

The WiFi Fingerprinting localization approach consists of two phases: i) training phase or commonly known as the offline phase that collects samples to build WiFi Fingerprinting dataset, and ii) the real-time phase of so-called the online phase that produces estimations based on incoming observations [2].

In terms of WiFi signal, indoor spaces experience a challenging radio propagation environment with multi-path effect, shadowing, signal fading and other forms of signal degradation and distortion. It is hard to model all the possible states of the WiFi environment in the offline phase, so often, the online phase experiences forms of the environment which never trained on, leading to erroneous estimations.

The main challenge of taking advantage of these sensor data is that erroneous signal silencing occurs during data collection and real-time data sampling phases.

Figure 2 shows the complexity of WiFi samples through the histogram of a long scan at a random indoor location. Although many WiFi-based positioning systems model RSS as a Gaussian process [17,18], we can see that none of the 5 AP histograms fits a normal distribution tightly. In fact, AP2 shows a bimodal distribution; AP4 and AP5 are skewed to the right and are overlapped in RSS that is likely to interfere with each other if on close channels. Whereas AP1 has a wide distribution of observed RSS values, spanning almost 20 dBm. The difficulty of modelling WiFi environments is also exposed in [4].

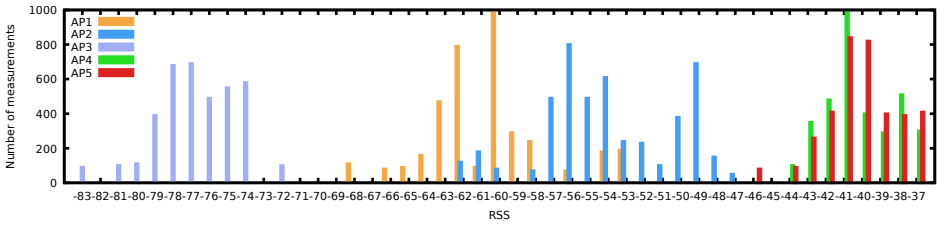


Figure 2. Histogram of Received Signal Strength for 5 Access Points observed at a single location showing the complexity of WiFi fingerprints, with various distributions (binomial and skewed).

We sampled the RSS from one AP at a fixed location. Figure 3 shows the fluctuation characteristic of APs strength. Each of these histograms is drawn from one APs at a fixed location throughout a short period of time. Erroneous signal silencing occurs irregularly. Any slight change in the environment hinders accurate estimations. Fitting a single polynomial density function to capture this wide variation of these histograms is hard to achieve.

Unlike mathematical-based solutions, using neural networks to torrent these noisy data as specific characteristics of particular locations, rather than simply cancel out these observations as noise or outliers. Hence, a model should assimilate information from new data easily and capture more unexpected variations. Others use deep neural networks in WiFi fingerprints signal strength based indoor localization [19], and also on WiFi signals with a formulation of the propagation model known as EZ [20,21], while more recent work has been using neural networks on Channel State Information (CSI) [22].

2.3. Multimodal Approaches

Multimodal approaches make estimations from multiple perspectives of cross-modality data. Filtering methods like particle filters and Kalman filters have been proposed to address multimodalities of data. Specifically, HiMLoc uses particle filters to integrate inertial sensors with WiFi fingerprints based on prior observations of Gaussian processes for direction estimation, distance estimation and correlation between samples

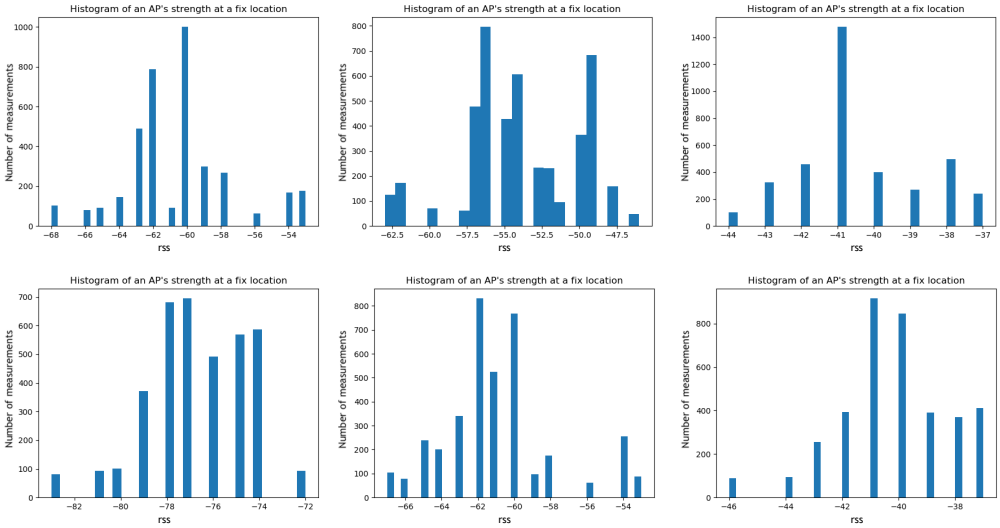


Figure 3. Histograms of one AP signal strength showing the fluctuating nature of WiFi signal, which makes this hard to model with simple function fitting. The x-axis represents received signal strength while y-axis means sensed AP count.

and location in buildings, and admissible human activity [2]. Similarly, WiFi-SLAM and Zee build on particle filters emphasising their importance for random system initialisation [3], while Kalman filters are used to integrate inertial sensing modalities [5]. Other engineered approaches, such as UnLoc, combining sensing modalities based on empirical observations of how some locations are unique across one or more sensors [6], MapCraft uses conditional random fields [23], LiFS uses graph constraints to map and position estimations on the trajectory [4]. Similarly, WILL builds a connected graph to estimate location at room level [24].

However, when samples are formed in multimodalities, solutions building on multimodal deep neural networks show their advantages of understanding correspondences between communicative multimodalities and capturing in-depth features from natural representations instead of focusing on a single modality without alternative feature inputs.

To date, multimodal neural networks across sensing modalities have not been used for indoor localization, although it has been used for other context recognition tasks, such as human activity recognition [8]. Here, we aim to customise an end-to-end multimodal deep neural network for the indoor localization task to produce location estimation based on sensor-fusion data of inertial movement measurements and WiFi fingerprints. Training directly on data has its drawbacks, that of moving the challenges to the quality of training dataset and cross-sensor modality alignment, although this can be eventually automated by other systems such as vision-based systems [25].

3. Methodology

In this section, we introduce the methodologies that how to use recurrent neural network for pedestrian dead reckoning based on inertial sensor data and how deep neural network performs a location estimator using WiFi fingerprints, as well as our proposed multimodal deep neural network architecture, which fuses both single-modality data to deliver end-to-end positioning regression outputs.

3.1. Pedestrian Dead Reckoning with Recurrent Neural Networks

PDR estimates continuous location by starting from assuming a known position and estimating displacement and direction of movement to estimate consecutive locations. By similarity, Recurrent Neural Networks (RNN) perform the same but with the advantage of memorising previous steps and not relying entirely on new observations coming into

the system, which can be affected by local noise. RNN is one of the artificial neural networks that include connections between nodes that flow along a sequence. This structure offers the RNN model has the ability to exhibit temporal information through time serial data.

RNN has proven its advantages in dealing with sequential data such as speech recognition, image captions and machine translation tasks [26]. RNN is similar to a feedforward neural network. The difference is that the recurrent connections link a neuron from the current layer to the next neuron of the next layer. This feature makes the RNN model 'remember' the features from the previous loop [27]. RNN transfers the state within each loop. Therefore, it could deal with sequential data such as the inertial sensor data we take used in the experiment.

The RNN structure is shown in Figure 4. It is an unfolded basic RNN structure that

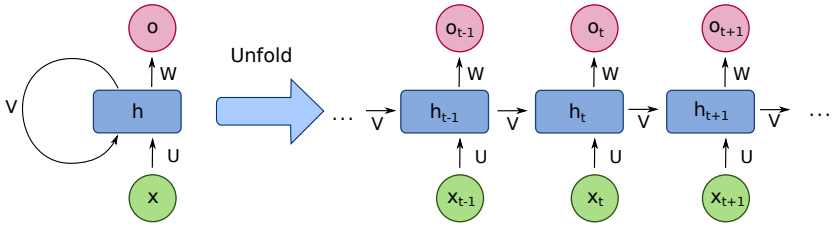


Figure 4. Fully Recurrent Neural Network Structure

contains a number of neuron-like nodes. Those connected nodes, which are either input, output or hidden nodes, are organised into successive layers which follows a one-way direction connected to the next layer. Each neuron includes an activation that varies based on the time sequence. Errors are calculated from each sequence, while the total error is the total value of the deviations of the target label values calculated from each sequence.

However, the basic RNN has the problem of vanishing gradient when feeding long sequential data. It cannot catch the feature of the dependencies between samples in relatively longer sequential data. Hence, Long short-term memory (LSTM) is presented to solve this problem. It performs similarly as Dead Reckoning to estimate positions from streaming inertial sensor data [28].

LSTM is an optimised RNN model that solves the basic RNN problem of vanishing gradient. This is achieved by adding a forget gate. It prevents vanishing or exploding caused by backpropagated errors. Figure 5 shows the internal structure of the LSTM unit. In each unit, there is not only an input gate, an output gate but also a forget gate that controls the 'memory' to either propagate into the next layer or being forgotten in the current layer [29].

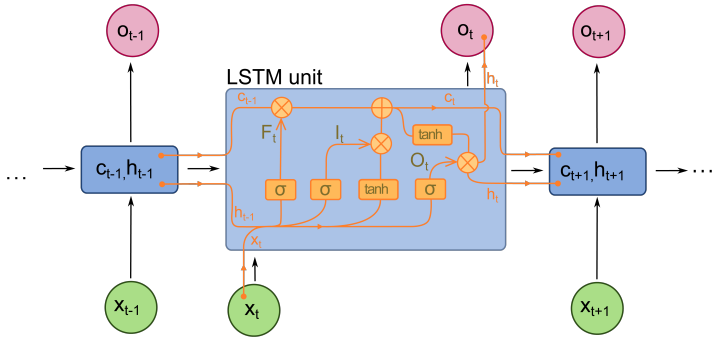


Figure 5. Long Short Term Memory (LSTM) Architecture

The value in the current state is controlled by the forget gate *f*. Specifically, save the value when the signal is set as *1* while forgetting the value if the gate is set as *0*. The activation of receiving a new input or propagating is determined by its input gate

and output gate, respectively [30]. The equations 1 to 6 show the numerical definitions. The x_t represents the input data at time t while W_i, W_f, W_o contain, respectively, the weights of the input and recurrent connections. Every time step t, LSTM calculates input(I_t), forget(F_t) and output gates(O_t) activation vectors, which decide the cell state value(c_t) and hidden cell(h_t). The softmax output of m_t determined the final probability distribution while $\odot$ is the product of the cell value of the gate value.

$$I_t = \sigma(W_{ix}x_t + W_{im}m_{t-1}) \tag{1}$$
$$F_t = \sigma(W_{fx}x_t + W_{fm}m_{t-1}) \tag{2}$$
$$O_t = \sigma(W_{ox}x_t + W_{om}m_{t-1}) \tag{3}$$
$$c_t = F_t \odot c_{t-1} + I_t \odot h(W_{cx}x_t + W_{cm}m_{t-1}) \tag{4}$$
$$m_t = O_t \odot c_t \tag{5}$$
$$p_{t+1} = \text{Softmax}(m_t) \tag{6}$$

Figure 6 illustrates the unrolled chain of the LSTM network, where C_t is the long-term memory at time t and h_t is the block output at time t, or short-term memory, both transmitted to the following LSTM block in the chain.

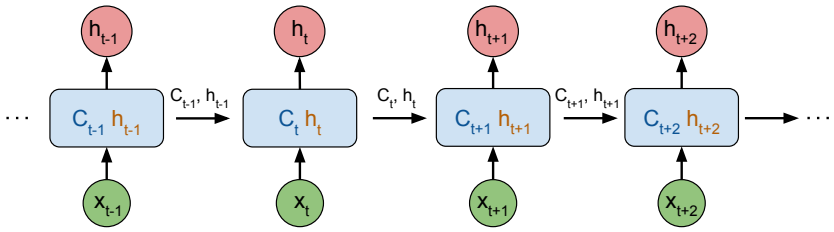


Figure 6. Unrolled Chain of LSTM Neural Network

As the sensor data is presented in time sequence, the LSTM model is ideal for location estimations. LSTM read the time-sequential inertial sensor data based on a fixed size of ($Timestep * Features$). The feature of each data point is the magnitude value of acceleration, gyroscope and magnetic field data. The number of data points in each sample is determined by the chosen time window (here, we set the time window as one second sampled every 100 milliseconds, explained later). Each sample is offered a target position in coordinate (X_i, Y_i). The regression output of the LSTM model is the estimated position in coordinates (X_{est}, Y_{est}) based on the history information and current sensor sampling input. The numerical definition is shown in equation 7 to 9.

$$x_{-1} = \text{Sensor Data}(I) \tag{7}$$
$$x_t = W_e S_t, \quad t \in \{0...N-1\} \tag{8}$$
$$p_{t+1} = \text{LSTM}(x_t), \quad t \in \{0...N-1\} \tag{9}$$

During the training process, LSTM updates its weight and state by each time reading a new incoming sensor data and returns a location estimation result. Here, x_{-1} represents the inertial sensor data from the last time step; x_t indicates the current stage LSTM block input of the weighted (W_e) sensor data S_t; p_{t+1} means the LSTM cell next-timestep numerical prediction of latitude and longitude based on the current stage information (x_t).

3.2. WiFi Fingerprinting with Deep Neural Networks

Indoor positioning systems use an anchoring mechanism to estimate the location based on instant snapshots of the environment sampled by sensors for recalibration. One such independent estimation can be achieved with WiFi Fingerprinting, which are regarded as unique at some positions [6] for locationing without continuous sensing. For periodic recalibration, the WiFi is a reliable anchoring signal source, used extensively in

previous research [2,3,6,24] due to this relying on real-time observations to match on the fingerprint database or with a pre-trained model for position estimation.

Here, we introduce a way of using deep neural networks (DNN) for WiFi Fingerprinting location estimations. The configuration of modelling WiFi Fingerprinting with DNN is shown in figure 7.

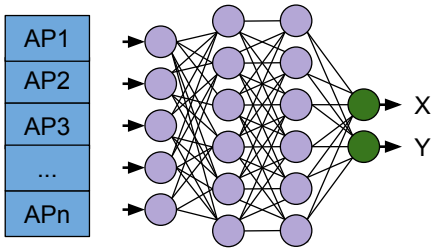


Figure 7. WiFi Fingerprinting Deep Neural Network

The end-to-end DNN takes WiFi scans from sensed access points at each sampling timestep as input features and (X_i, Y_i) coordinate of where the fingerprint is collected as target variables during the training phase while producing two numerical outputs of (X_{est}, Y_{est}) for each coordinate as geographical location estimations during the online inference phase.

3.3. Sensor Fusion by Multimodal Deep Neural Networks

By fusing both inertial sensors and WiFi fingerprints modalities, their unique perspectives can contribute to more robust estimations. Similar to our previous work in multimodal deep learning for context recognition [8], here we explore the capacity of similar construction to combine the two aforementioned neural networks operating on each sensing modality.

Figure 8 presents our proposed MM-Loc architecture, an end-to-end multimodal deep neural network for performing indoor localization, which makes inference on the joint perspectives of inertial sensors and WiFi fingerprints modalities.

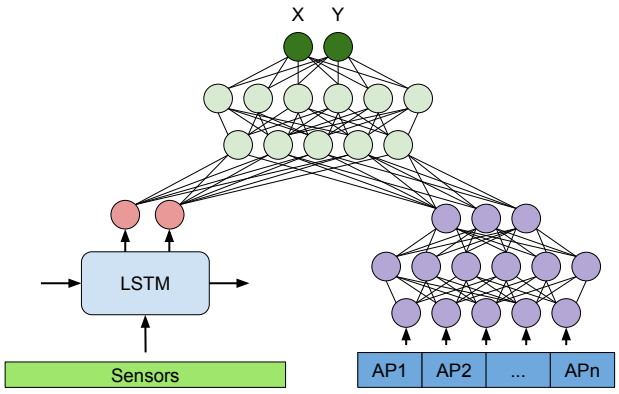


Figure 8. MM-Loc: our proposed multimodal deep neural network architecture for indoor localization with two parallel single-modality feature extractors, and a joint network structure to merge latent features at the top.

The MM-Loc first reads time-sequential inertial sensor data by LSTM sub-network and WiFi RSS data by DNN sub-network synchronously from the beginning, where the sample size of the sensor modality at the LSTM side is $(Timestep * Sensor\_num)$ and for the DNN side is $(Timestep * AP\_num)$. Both modalities are reshaped to 128-dimensional hidden outputs from each branch. These two parallel 128 unit hidden outputs are then fused by concatenation to 256 units. The 256-dimensional joint vectors are fed into three fully connected (FC) layers with the input size of 256, 128 and 64, then transferred to

the top prediction layer. This operates as a regression, producing location estimations in two-dimensional outputs of (X_{est}, Y_{est}) from the last joint cross-modalities hidden layers.

The contribution of LSTM to the cross-modality component is in the form of its multiple LSTM layers worked as feature extractor, which passes sensor hidden feature towards the higher layer. The DNN component contributes as a WiFi feature extractor. We replace the two units regression outputs, originally from modality-specific neural networks, by fully connected layers to transfer extracted hyper-dimensional vectors to joint layers. Both bottom neural network components pass the individual sensor modality hidden features synchronously to the cross-modality layers with element-wise calculation and transferred the fused latent information to the final layer for location prediction regression. The inference completes in one go that the end-to-end MM-Loc read the multisensory data as input and return the location estimations as outputs. During the training process, the MM-Loc containing three sub-components of LSTM, DNN and cross-modality neural networks are trained synchronously with computing the loss in the forward pass and splitting the gradients on each branch on the backward pass.

What is unique about this construction is its ability to handle either imbalanced multisensory sampling rates or missing inputs from the WiFi modality. This is because WiFi scans are produced at a much lower rate than inertial sensors, so when there is no WiFi scan available, the WiFi input is a vector with all components value of 0 (normalised to -100 dBm). In this situation, the WiFi branch of the network contributes negligibly to the cross-modality component, in which case the majority of the contribution comes from the inertial sensor modality branch. When both modalities have inputs, the same multi-modal architecture automatically adjusts the weights of cross modalities to produces estimations based on both modality contributions.

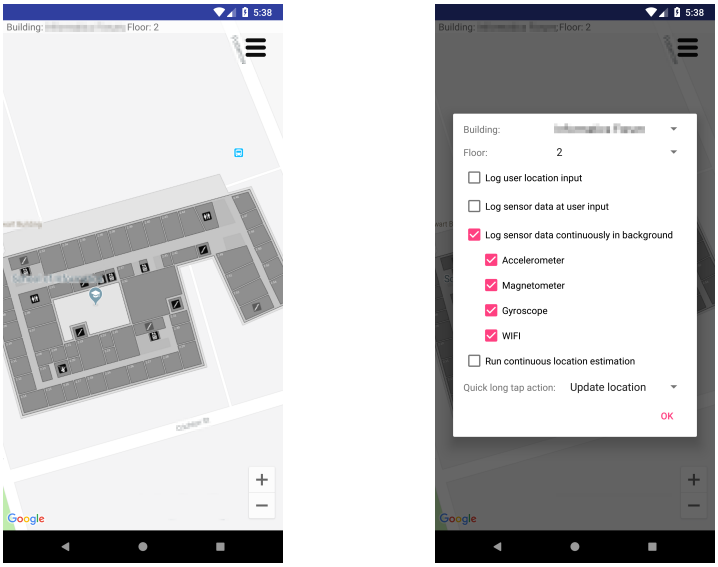
4. Data

In this experiment, we create the sensor fusion dataset by ourselves, starting from data gathering to data preprocessing and eventually aligning multimodalities to build a sensor fusion training dataset. The cross-sensor datasets are collected from two scenarios with ground truth location labels. With light data processing strategies of interpolation, normalisation, overlapping, and down-sampling, we derive a uniform machine learning dataset for training the proposed multimodal deep neural network.

4.1. Data Collection

Multisensory data are collected using an Android application designed specifically for the task of data collection, screenshots shown in figure 9. This application can be configured to collect sensors data (accelerometer, magnetometer, gyroscope and WiFi scans) continuously in the foreground with a visual interface to accept user inputs or in the background mode to allow carrying the phone in a pocket with the screen off. To collect data with ground truth location information, the same application runs synchronously on two mobile phones. One used to mark locations as input from a user through the visual interface displaying the building map aligned to Google Maps coordinates for the purpose of longitude and latitude information acquisition. It needs to mention that the logged ground truth geographical locations are converted to the Universal Transverse Mercator (UTM) in metres in the experiment. Another phone is used to collect sensor data continuously in the background, which is placed in any position without the need for user interactions while collecting data to resemble the perspective of sensors in a natural motion.

During the data gathering process, ground truth labels are transferred to the device, collecting multiple sensor data to annotate sensor samples with the user-provided location of their provenience. The first campaign involved walking on the corridor at different walking speeds, starting from one corner of the building, performing a circular trajectory and arriving back at the starting point. Ground truth locations were provided



(a) Location Input Interface (b) Sensors Control Options

Figure 9. Screenshots of the Android application used to collect multisensory data.

sporadically, but to obtain location information on a more granular basis, some locations were interpolated between consecutive two input locations – assuming local constant walking speed. Specifically, during one round of data collection, we build two datasets synchronously: Dataset.1 that walking along the corridor to collect samples from inertial sensors and WiFi scan; Dataset.2 that collecting ground truth geographical location labels synchronously by clicking the screen to log latitude and longitude information when passing key locations such as corners.

For training our MDNN model with generalisation, we collect data from two scenarios to build the dataset – the middle floors from two crowded office buildings. Both scenarios are typically indoor environments with multiple human activities to increase indoor complexity in terms of noisy data and scalable description. Specifically, these complex situations include people walking by frequently that affect how we walk along the corridors during data gathering; various indoor working electronic devices such as elevators, computers, printers and portable devices which generate electromagnetic radiation; Building materials contain reinforced concrete, metal and glass which influence signal's propagation pathway. Furthermore, considering the real-time situation that people using different hardware for locationing purposes, we use multiple mobile devices with different built-in sensor sensitivity and sampling rates for data gathering to add complexity. During data collection, we walk with different speeds and gestures to add motion variations of the samplings.

By integrating 14 rounds of sampling files collected by ourselves, we derive the MDNN datasets from two scenarios, illustrated in table 1.

Table 1. MDNN Dataset Description

Datasets	Inertial Samples	WiFi Samples	Access Points	Time Duration
Scenario A	24450	25541	102	407 Mins
Scenario B	29836	8390	750	497 Mins

4.2. Data Preprocessing

After sampling multisensory data, we need to maintain the original features but process the logged data into a format that is compatible with machine learning training and inference.

4.2.1. Inertial Sensor Data

Using the Android APP, we export the raw log files of sensor fusion data from two scenarios. However, the Android API provides sensor samples on an event base only if the value is considered to have changed. This leads to uneven intervals in each sensor sample. Inertial sensors of accelerator, gyroscope and magnetometer hold different refreshing rates. Here, we implement linear interpolation to fill the missing values in between every two hardware sensed values in order to generate the continuous time-sequential data. These interpolated data are grouped in a time window, which we discussed later, and associated with one location. We also interpolate geographical locations based on the selected time window between each two recorded locations as we only record locations when passing special points such as corners, elevators, kitchen, meeting rooms and other remarkable places along the trajectory.

The inertial sensors sample the motion in three orientations of axes x, y, z. To allows for the phone to be placed in any orientation in a pocket or a bag without the limitation of motions and fixed positions. Position invariant sampling data is attained by calculating the magnitude value on the built-in sensors (accelerometer, gyroscope and magnetometer) on the three orthogonal axes:

$$sensor_{magnitude} = \sqrt{sensor_x^2 + sensor_y^2 + sensor_z^2} \tag{10}$$

Hence, we derive the magnitude values of inertial sensor samplings used for the machine learning time-sequential dataset.

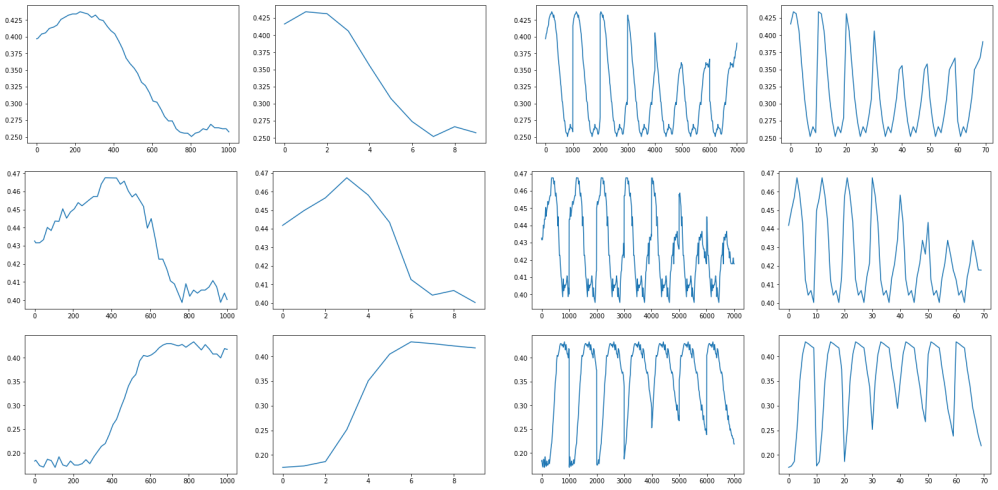
As the LSTM load the time-sequential data by time window, a proper window selection contributes to lower computational cost and on-device power consumption with accepted inference efficiency. We explored four time windows settings of 10ms, 100ms, 1,000ms and 2000ms. In addition, we considered the case of time windows overlapping (10%, 50%, 90%) to increase the frequency of location updates for a fast responsive system. Window overlapping also plays well with the LSTM model since the information from previous time windows is reinforced and emphasised by overlapping for better quality detection and strengthen correspondences between samples.

Despite the amount of machine learning dataset increased with richer information for training by time window grouping and overlapping, the side effect is that heavier dataset increases the processing demand. Hence, to improve the feed-forward speed with a smaller input size without losing too much information, a down-sampling method is investigated to compress the dataset. Figure 10(a) shows three magnitude values for accelerometer, gyroscope, magnetometer with 1,000 data points (1,000ms) on the left side and the down-sampled value to as much as 90% linear compression. Figure 10(b) compares the original data and the down-sampled over a longer interval of 7,000ms. The comparison from figure 10 indicates that main features are maintained even on high compression, reducing samples frequency to 1 Hz. For this reason, we consider down-sampling in experiments presented in the evaluation section.

Therefore, by deploying data processing of time window grouping, overlapping and downsampling, we derive the ready-to-use inertial sensor dataset with the settings of 1 second time window, 90% overlapping rate and 90% downsampling compression data preprocessing strategies.

4.2.2. WiFi Fingerprint Data

The RSS inputs to neural networks are provided as vectors. To construct this vector, we first scan the whole WiFi logfile to identify all unique APs observed throughout the data gathering section (total of n APs observed inside the building), as well as the minimal and maximum signal strength encountered throughout, which are used to normalise the vector input to [0,1] interval by linear scaling. By observation, the min-max interval is [-100,-40] in dBm. Hence, for missing APs in WiFi scans, a value of -100 is associated with their representation in the n-dimensional vectors. To keep



(a) Sensor values over a time window of 1s (b) Sensor values over a wider time window of 7s

Figure 10. Compare between the original data and the down-sampled data. It shows that loss of information is minimal across two time windows, being able to follow the trend in signal for the walking activity.

the original features of the sampled data without unnecessary human intervention, we keep those occasionally seen personal hotspots, considered as noise, in the dataset to add complexity, which simulates the real environments of changeable WiFi signal distribution.

4.3. Sensor Fusion Dataset Alignment

The main challenge of aligning inertial sensor data with WiFi data is that the WiFi sampling frequency is significantly slower than the inertial sensing rate due the hardware limitations.

By observing the original refreshing rate (on the event base) from log files, the average updating frequency of the inertial sensor is in milliseconds while WiFi updates in seconds. (It is noticed that the time steps of WiFi samples recorded naturally in non-integer values on event base, such as 1.03s, 3.56s, etc.). The amount of the time-series sensor data is significantly larger than WiFi samples, even after we group the inertial sensor data by one second time duration as one sample.

If we simply combine these two modalities (one-second grouped inertial sensor data with one WiFi scanning as a cross-sensor datapoint input), the cross-sensor dataset contains sparsity with unbalanced data components. As the fact of the WiFi scan updates on the event base with the waiting time gap over several seconds. For the purpose of eliminating the sparsity of the WiFi data to make it relatively compatible with the grouped sensor data, we squeeze the WiFi scans from the original sampling distribution to every 100 milliseconds time duration, which is compatible with the LSTM time window setting. For instance, if the first time duration of 0.1 seconds in the raw log file contains three WiFi scans through all accessible APs, we compress these three samplings into one single sample within the time duration. Therefore, we get a denser WiFi fingerprint dataset to be aligned with grouped inertial sensor data, illustrated in table 2.

Table 2. WiFi Fingerprint Dataset Compression

Time	AP_0	AP_1	...	AP_n	X	Y
t_0	Null	−85	...	Null	x_0	y_0
t_1	−92	Null	...	Null	x_1	y_1
t_2	Null	Null	...	Null	x_2	y_2
$T_\prime$	−92	−85	...	Null	$X_\prime$	$Y_\prime$

For some timesteps, if there only contains inertial sensor samplings without any accessible WiFi signals, we use -100 dBm to represent missing WiFi scans of all APs if there is no sensed signal through all APs in one time window. They are added in parallel for a uniform size of the multimodal dataset.

As two synchronously-logged datasets contain not only inertial sensor and WiFi RSS samples but also ground truth location information within the same time duration, the timestep records are utilised for matching multimodalities with geographical labels. Meanwhile, location labels are normalised to the extreme boundaries chosen for the building and scaled to [0,1] intervals. Estimations of neural network models are converted back into latitude and longitude coordinates in meters to measure the estimation errors as the euclidean distance between predict and target locations. Table 3 illustrates the components of the cross-sensor dataset after alignment.

Table 3. Cross-sensor Dataset

Time	Accelerator	Gyroscope	Magnetometer	AP_0	AP_1	...	AP_n	X	Y
T_0	$a_0 \sim a_{999}$	$g_0 \sim g_{999}$	$m_0 \sim m_{999}$	-100	-85	...	-100	X_0	Y_0
T_1	$a_{999} \sim a_{1,999}$	$g_{999} \sim g_{1,999}$	$m_{999} \sim m_{1,999}$	-100	-100	...	-100	X_1	Y_1
T_2	$a_{1,999} \sim a_{2,999}$	$g_{199} \sim g_{2,999}$	$m_{1,999} \sim m_{2,999}$	-70	-100	...	-65	X_2	Y_2
...	...	...	...	...	...	...	...	...	...
T_n	$a_n \sim a_{n+999}$	$g_n \sim g_{n+999}$	$m_n \sim m_{n+999}$	-100	-100	...	-100	X_n	Y_n

5. Evaluation

This section presents the evaluation of each independent modality-specific neural network architecture, followed by the evaluation of multimodal deep neural network (MDNN) implementations with different fusion architectures that combining the features extracted by each independent model (RNN for sensor data and DNN for WiFi samples) to produce a single robust location estimation at the top.

5.1. Recurrent Neural Network on Inertial Sensors

Here we present our exploration to identify the best model structure and parameters settings for calibrating the LSTM with the best positioning performance in terms of time window settings, overlapping ratio and data compression.

5.1.1. Time Window

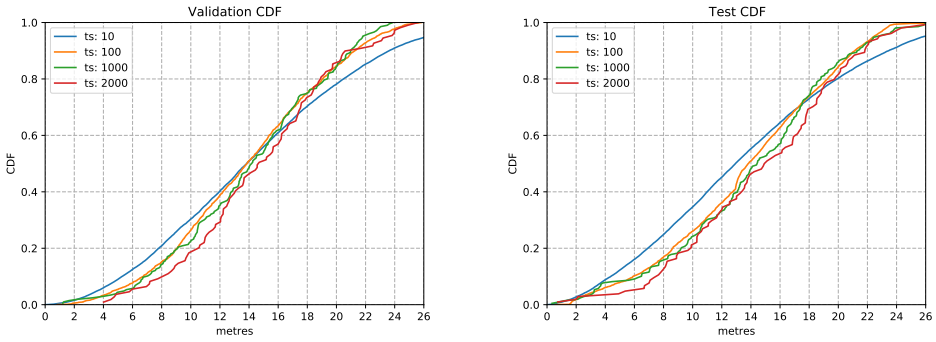
As LSTM read the data in time windows, a well-selected time window allows the model to catch enough detailed information to estimate the movement accurately. A larger time window loses granular details by exploiting larger scale observations, including more information for a range of movements while being computationally demanding for performing inference on mobile devices and slower in providing location updates. In contrast, a smaller time window captures minimal information, not discriminating between different walking speeds or between very similar activities like moving on a flat surface and climbing stairs, although more computationally friendly to mobile devices since the input layer is smaller.

Here, we trained the LSTM model with the time window of 10ms, 100ms, 1 second and 2 seconds, with the model setting listed in table 4.

Figure 11 shows that the LSTM model performances with various input time windows, presented using the Cumulative Distribution Function (CDF) charts. By observation, time window setting has little impact on the inference accuracy. The training

Table 4. LSTM Training Parameter Settings

Parameter	Settings
Epoch	100
Batch Size	100
Hidden Units	128
LSTM Layer	1 Layer
Learning Rate	0.005
Learning Rules	RMSprop



(a) Validation Set CDF (b) Test Set CDF

Figure 11. LSTM model performances with different time window settings.

observation is confirmed on the test set as well as seen in the CDF plot 11(b), showing that chosen time windows have minimal impact on detection accuracy. The 1,000ms based model shows a good performance similar to the others, and going for this larger time window allows the model to capture variations and different activities relevant when transitioning between different indoor areas (elevator, cafe). The estimation error is still high since this is the first parameter we optimise for. Therefore, we choose the time window setting of 1,000ms in the following explorations.

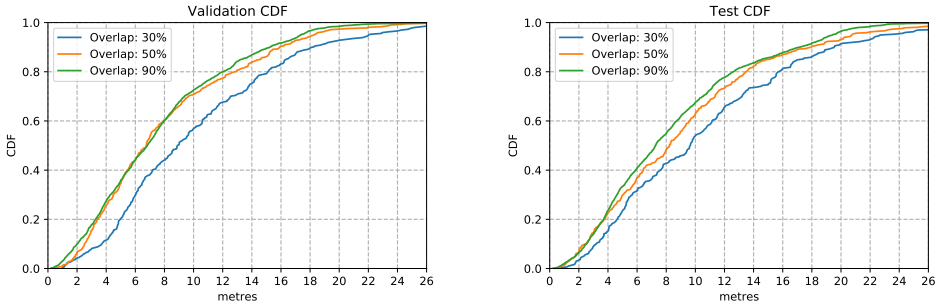
5.1.2. Overlapping Ratio

This evaluation presents the outcome of changing the overlapping ratio based on the fixed time window setting of 1,000ms. The overlapping ratios we experiment with are 30%, 50% and 90%, which increase the amount of training data subsequently by 1.3×, 2× and 9× respectively.

There are two main advantages to implementing overlapping. The first one is to enhance dependency between consecutive time windows samples by repeated information observed in the overlapping part. With the LSTM model, this enhances the memory aspect of adjacent time windows, the model experiencing portions of the recent action through multiple inputs. Moreover, we increase the training dataset synthetically by obtaining a larger number of training samples, since larger training sets are beneficial when training neural networks.

Figure 12 shows the three models training process with the enhanced training set and performance on the validation set. In the training set, it appears that the model trained with data overlapping by 90% performs consistently better than the other two models trained on 30% and 50% overlapping data. While in testing set CDF shown in figure 12(b), 90% overlapping model performs similarly as in the training set. This is an impressive result considering that it is based on nothing more than inertial sensor data without much to calibrate on apart from occasional well-located changes of direction (e.g., when going around a corner). This route covered long stretches of straight-line corridors (up to 60 meters) walked at various speeds, which conditions are identified

well by the LSTM estimator. These results illustrate that by increasing the overlapping rate, models can learn features better since they have more data available to train on.



(a) Validation Set CDF (b) Test Set CDF

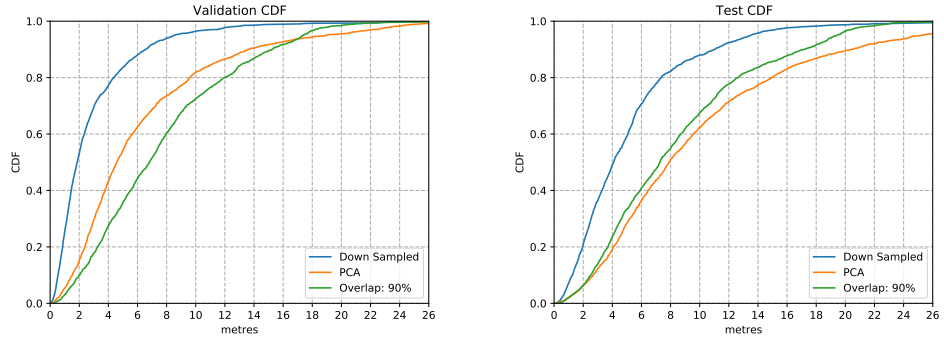
Figure 12. LSTM model performance with different overlapping rations on validation and test set.

From this experiment of using overlapping strategies, model performance improves compared to those without time windows overlapping. The 90% windows overlapping model has the best performance. We use this enhancement for the following model exploration.

5.1.3. Data Compression

We implement simple downsampling and Principal Component Analysis (PCA) strategies for data compression to speed up the training and inference time cost. As mentioned in figure 10, simply downsampling has little impact on the original signal representations. Hence, we implement downsampling (pick up datapoint every 100ms) over the training data with the overlapping ratio of 90%. Therefore, one sample has been downsampled from the size of $(1000 * 3)$ to $(10 * 3)$. For the PCA data compression, the dimension is reduced to the same size of $(10 * 3)$ of each sample. New variables in a lower dimension could be calculated based on eigenvalues and eigenvectors and therefore replace the original variables in a higher dimension matrix by PCA. This avoids having LSTMs a heavy time step input of 1,000 samplings which increases the model's complexity and learning time spent.

Figure 13 shows the comparison between the down-sampled based model and PCA based model with the uncompressed overlapping 90% based LSTM model. Both CDFs illustrate that the down-sampled based model performs better than the uncompressed data based model that it reaches 80% of the prediction accuracy with the precision requirement of 8 metres of the testing set. PCA based model performs slightly better compared to the uncompressed model in both validation and testing sets.



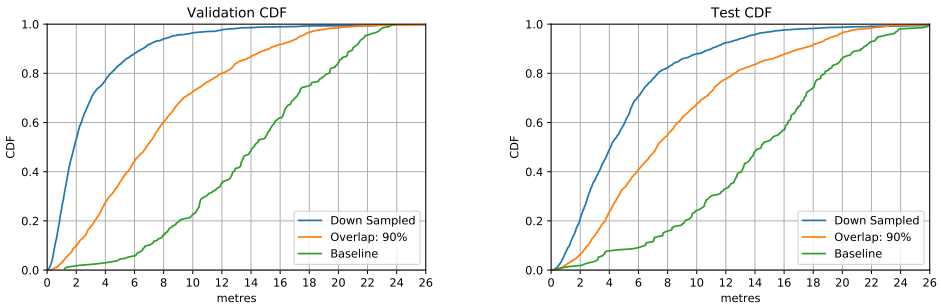
(a) Validation Set CDF (b) Test Set CDF

Figure 13. LSTM Model performances with different data compression strategies.

In general, the down-sampled based model has the highest accuracy and reliability compared to the PCA-based and the model without data compression. By implementing downsampling to the overlapping data, the model reduces its complexity by setting a smaller time step to save calculation time. It could handle the data with a larger time window without losing significant data features. Specifically, ten datapoints in one time window represent the data features of the one-thousand samples in a time window, which allows the model to further increase the time window allowance with little model complexity increased. It could potentially allow the model to process a variety of activities. Besides, it widens the model's tolerance for a more extensive training dataset in a shorter time span compared with feeding the uncompressed data. This is of vital importance when integrating the model into mobile devices to improve the prediction efficiency, which reduces application response time and power consumption.

5.2. Overview of RNN for Inertial Sensors

Figure 14 contains all models discussed in this section. Overall, Starting with selecting a suitable time window from 10ms, 100ms, 1 second and 2 seconds, we choose 1 second as the time window as it allows more data variations. We are moving to improve estimation accuracy by further increasing the training size with overlapping. With 90% of the overlapping, the model shows a significant improvement in locationing accuracy. To reduce the complexity and the training time of the models, we compress the training data size by downsampling and PCA dimension reduction. We selected the LSTM model with downsampling based on a 90% overlapping ratio, presented by the blue line here, as the inertial sensor locationing model settings, which balances the estimation accuracy and efficiency.



(a) Overall Validation Set CDF (b) Overall Test Set CDF

Figure 14. Overall comparison of model performances with different calibration strategies.

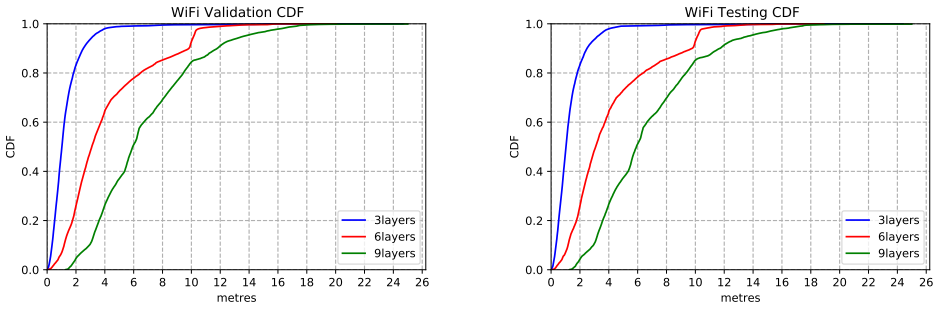
5.3. Deep Neural Network on WiFi Fingerprints

WiFi scans are received through the Android API at an irregular frequency on the event base, which its average update rate of about one second. We used Deep Neural Network (DNN) as our configuration of the WiFi Fingerprinting model, which takes WiFi scans from sensed access points at each sampling timestep as input features and produces two numerical outputs of latitude and longitude as estimations. We evaluate the WiFi estimator regarding model structure settings and tuning based on the processed WiFi fingerprint dataset. The missing WiFi samples represented by -100 dBm are converted to zero values using normalisation before feeding into the model.

5.3.1. Model Structure

We explore the impact of DNN model structures of 3-layers, 6-layers and 9-layers settings. As we observe from the CDF results presented in figure 15, the 3-layer DNN regression model produces the best inference accuracy than the deeper networks. The

models show extreme similar estimation results in the testing CDF 15(b), which approves the ability of model generalisation.



(a) Validation Set CDF (b) Test Set CDF

Figure 15. WiFi based DNN model performances with different network structures.

5.3.2. Model Tuning

After we determine the DNN structure, we implement model tuning on the 3-layer WiFi DNN regression model with the hyperparameter settings shown in 5. It is mentioned that the only variation of the model structure is the input sizes caused by the number of APs sampled in different experiment buildings. For our two evaluation scenarios, there are 102 APs and 750 APs, respectively.

Table 5. WiFi-based DNN Parameter Settings

Parameter	Settings
Input Size	AP Number
Epoch	100
Batch Size	100
Hidden Units	128
DNN Layer	3 Layers
Dropout Rate	0.5
Learning Rate	0.001
Learning Rules	RMSprop

5.4. Overview of DNN for WiFi Fingerprints

The WiFi-based estimator is modelled with a deep neural network regression model. By exploring the model structure with parameter tuning settings, we determine to use a three-layer DNN model on WiFi Fingerprinting. We use this WiFi model architecture as the sub-component network integrated into the multimodal fusion model.

5.5. Multimodal Deep Neural Networks on Sensor Fusion

After we determined the model structure for each modality, we move our focus on fusing each modality-specific network component representation together for a uniform multimodal deep neural network (MDNN).

5.5.1. MDNN Integration

To explore the best fusion architecture of the MDNN that links the modality-specific features extracted from the RNN and DNN sub-networks as introduced in figure 8, we customise four types of fusion networks, including two hybrid element-wise fusion of concatenation and multiplication, a hybrid residual connection fusion as well as a late fusion structure.

Element-wise Fusion: The MDNN with element-wise fusion architecture is shown in table 6. By concatenating the modality-specific hidden layer outputs from both LSTM

and DNN sub-networks of 128 dimension output, the fusion layer read these two hidden outputs by implementing element-wise matrix calculation of concatenation ($128 * 2$) or multiplication (128). This fused matrix then goes through higher 128 and 64 size fully-connected joint layers and eventually be regressed to two location prediction outputs of (X_{est}, Y_{est}).

Table 6. MDNN Architecture with Element-wise Fusions

Layers			**Output Shape**
LSTM Layer (sensor)			(Batch Size,128)
FC Layer.1 (WiFi)			(Batch Size,128)
Dropout Layer.1 (WiFi)			(Batch Size,128)
FC Layer.2 (WiFi)			(Batch Size,128)
Dropout Layer.2 (WiFi)			(Batch Size,128)
FC Layer.3 (WiFi)			(Batch Size,128)
Fusion Layer (concat or multiply)			(Batch Size,128*2 or 128*1)
FC Layer.4 (joint)			(Batch Size,128)
FC Layer.5 (joint)			(Batch Size,64)
FC Layer.6 (joint)			(Batch Size,2)
Batch Size	**Learning Rate**	**Learning Rules**	**Dropout Rate**
100	0.001	RMSprop	0.5

Residual Connection Fusion: Table 7 shows the MDNN with a residual connection architecture. Different from the element-wise fusion MDNN, in order to emphasise the WiFi feature input as its information amount is relatively incompatible with that of time-sequential inertial sensor data, we add a residual connection layer that transfers the hidden output (128) from WiFi penultimate fully-connected layer to the joint layer, fusing together with the LSTM (128) and DNN last FC layer outputs ($128 * 2$). So far, we derive a $128 * 3$ representation for the sensor-fusion component, which performs the final location estimation.

Table 7. MDNN Architecture with Residual Connection Fusion

Layers			**Output Shape**
LSTM Layer (sensor)			(Batch Size,128)
FC Layer.1 (WiFi)			(Batch Size,128)
Dropout Layer.1 (WiFi)			(Batch Size,128)
FC Layer.2 (WiFi)			(Batch Size,128)
Dropout Layer.2 (WiFi)			(Batch Size,128)
FC Layer.3 (WiFi)			(Batch Size,128)
Residual Layer (FC Layer.2 WiFi)			(Batch Size,128)
Fusion Layer (LSTM, FC Layer.3, RL)			(Batch Size,128*3)
FC Layer.4 (joint)			(Batch Size,128)
FC Layer.5 (joint)			(Batch Size,64)
FC Layer.6 (joint)			(Batch Size,2)
Batch Size	**Learning Rate**	**Learning Rules**	**Dropout Rate**
100	0.001	RMSprop	0.5

Late Fusion: Table 8 presents the MDNN with late fusion architecture. By combining two separate LSTM and DNN models' outputs, we build additional neural network layers on top of the predictions that produces the lat-long coordinate estimation (X_{Sensor}, Y_{Sensor}) and (X_{WiFi}, Y_{WiFi}) respectively. These estimations form a four-dimensional feature input space, which provides the representations needed by the top layers to estimate the final latitude and longitude (X_{Fusion}, Y_{Fusion}).

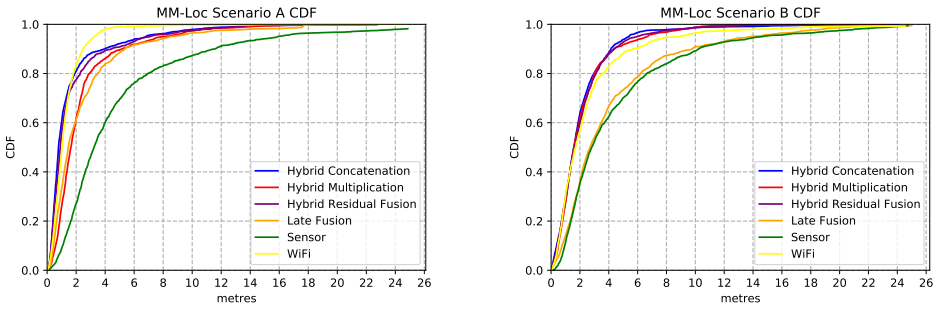
5.5.2. MDNN Implementation

We illustrate the estimation results from four types of MDNN with different fusion strategies, comparing sensor and WiFi single-modality location estimators results. CDF charts are shown in figure 16. We evaluate these models on the aligned multimodality

Table 8. MDNN Architecture with Late Fusion

Layers		**Output Shape**	
LSTM Layer (sensor)		(Batch Size,128)	
Sensor Regression Output.1 (X_1, Y_1)		(Batch Size,2)	
FC Layer.1 (WiFi)		(Batch Size,128)	
Dropout Layer.1 (WiFi)		(Batch Size,128)	
FC Layer.2 (WiFi)		(Batch Size,128)	
Dropout Layer.2 (WiFi)		(Batch Size,64)	
FC Layer.3 (WiFi)		(Batch Size,32)	
WiFi Regression Output.2 (X_2, Y_2)		(Batch Size,2)	
Fusion Network (input:X_1, Y_1, X_2, Y_2)		(Batch Size,2*2)	
Regression Output.3 (X_3, Y_3)		(Batch Size,2)	
Batch Size	**Learning Rate**	**Learning Rules**	**Dropout Rate**
100	0.001	RMSprop	0.5

dataset collected from two buildings (deployment scenarios) with the following split radio: 65%, 25% and 10% for training, validation and testing, respectively.



(a) Comparison CDF on Scenario A (b) Comparison CDF on Scenario B

Figure 16. Comparison of MDNN model performances with different fusion architectures.

Figure 16(a) shows the performances of the hybrid concatenation, hybrid multiplication, residual connection MDNN and late fusion MDNN, as well as the sensor model and WiFi model on scenario A. It is noticed that the hybrid concatenation fusion MDNN performs the best with 1.98 metres precision within 80%, following by the residual fusion and multiplication fusion models. Although the late fusion model has a relatively poor 3.7 metres accuracy, it is approximately 2× better than that from the sensor-based estimator. In terms of single-modality estimators' performances, it is noticed that the WiFi model perform significantly better than the sensor model with 2.6× better accuracy. Within estimation accuracy of 80%, the WiFi model has a precision of 2.6 metres error while the sensor model holds 6.9 metres prediction error.

Based on the evaluation results over scenario B, we observe similar performances from these models with the best performance contributed by the same architecture of concatenation fusion MDNN shown in figure 16(b). Regarding mono-modality model performances, the WiFi model has an error of 3 metres, and the sensor-based model has an error of 6.2 metres.

5.6. Overview of MDNN on Sensor Fusion

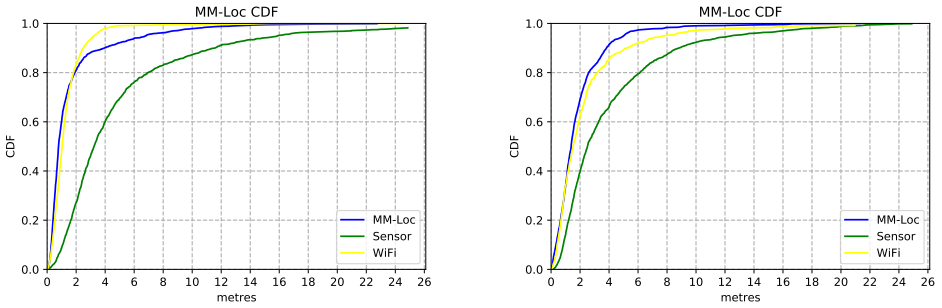
By evaluating the performance of MDNN under different fusion architectures (concatenation, multiplication, residual connection and late fusion), we observe that the median error of the hybrid fusion multimodal model in the testing set is only 1.98 meters, which is significantly lower than other fusion models and modality-specific estimators of sensors and WiFi data. In general, the CDF plot in Figure 16 shows that 90% of the errors are lower than 4 meters in both scenarios. The hybrid concatenation MDNN fusion method has the best estimation accuracy among all other models under

both scenarios, which gives us confidence in the generalisation power of the models. We determine to use the element-wise concatenation as the fusion method of the MDNN, given name as MM-Loc, for further evaluations.

6. Results

In this section, we evaluate the MM-Loc performance under both scenarios, compared to modality-specific models performances. Meanwhile, we discuss the inference accuracy influenced by sensor inputs with different sampling frequencies. Furthermore, we add a comparative study on evaluating our proposed MM-Loc with other state-of-the-art multimodal positioning solutions based on our self-collected datasets, followed by a trajectory visualisation result predicted by the MM-Loc.

6.1. MM-Loc Evaluation



(a) MM-Loc Performance CDF on Scenario A (b) MM-Loc Performance CDF on Scenario B

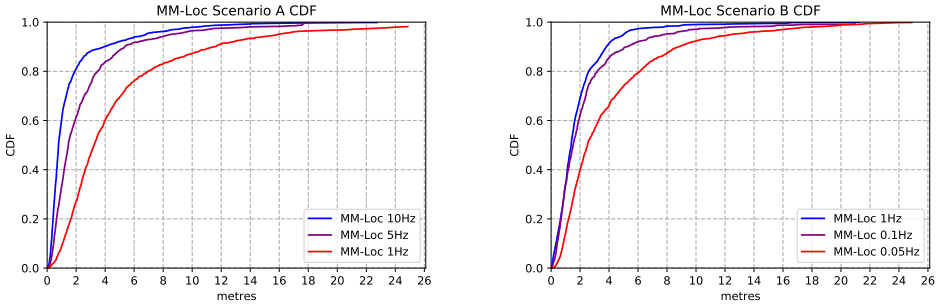
Figure 17. MM-Loc prediction CDF compared with modality-specific models.

In general, MM-Loc performs best in both scenarios. MM-Loc's median accuracy is within 2 metres error for 80% of the prediction cases, which is 3.5× better than the single-modality baseline model. By comparing figure 17(a) and 17(b), we observe that in scenario A, it is interesting that the MM-Loc performs better than the WiFi based single-modality model within the precision of 2 metres, at the intersection point with the WiFi model. However, the WiFi model outperforms the MM-Loc for larger precision tolerance. In scenario B, MM-Loc always outperforms the single-modality models. This is likely due to the WiFi signal coverage is denser in scenario A than that in scenario B, which contribute to the WiFi based position estimators. In general, within a strict accuracy expectation, MM-Loc outperforms other single-modality models.

To explore the opportunity to reduce energy consumption, we vary the WiFi scan frequency in our proposed MM-Loc system. Specifically, for scenario A, as the default WiFi sampling rate is 10Hz, sourced from the system, we reduce the scanning frequency of the dataset from 10Hz to 5Hz and 1Hz by applying a filter. The purpose of adjusting WiFi frequency is to assess the impact of this energy-saving strategy with scanning reduction on the location estimation accuracy. This also shows how our model behaves in systems where a high refresh rate is not available. For Scenario B, we decrease the WiFi sampling frequency from the original 1Hz to 0.1Hz and even 0.05Hz for the same reason.

Figure 18 presents the comparison between the performances of MM-Loc running at different WiFi sampling frequencies. We found that the multimodal model prediction accuracy experiences the same trend with decreasing sampling rates. MM-Loc with intermediate refreshing rate data still predicts with approximately 4 meters precision. Hence, a proper sampling rate setting has minimal contribution to on-device computing cost and power consumption with reliable positioning accuracy.

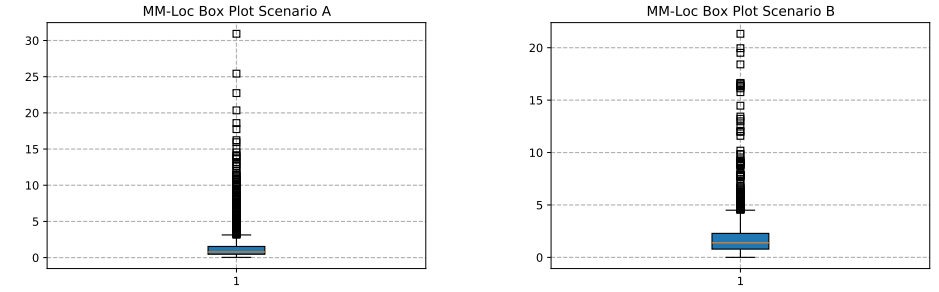
Figure 19 shows the box plots of the MM-Loc with the default WiFi frequency inputs under both scenarios. In scenario A, the accuracy at first quartile (Q_1), second



(a) MM-Loc Performance CDF on Scenario A (b) MM-Loc Performance CDF on Scenario B

Figure 18. Comparison of MM-Loc model performances running at different WiFi sampling rates.

quartile (Q_2) and third quartile (Q_3) are 0.467, 0.784 and 1.545 meters respectively; While in scenario B, Q_1, Q_2 and Q_3 are 0.789, 1.387 and 2.288 metres respectively.



(a) MM-Loc Box Plot on Scenario A (b) MM-Loc Box Plot on Scenario B

Figure 19. MM-Loc Prediction Box Plot

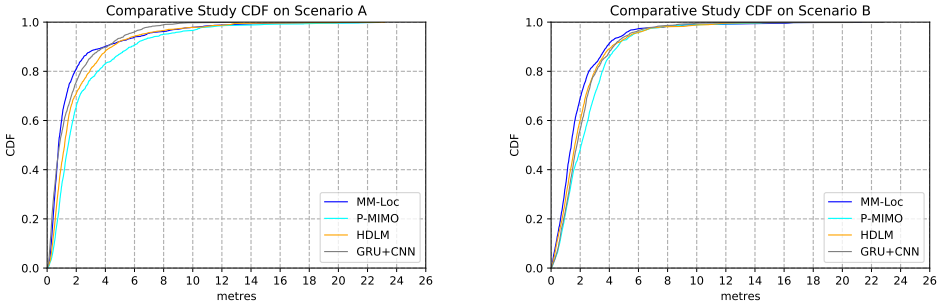
6.2. MM-Loc Comparative Study

We compare our proposed MM-Loc performance with other multimodal state-of-the-art (SOTA) positioning systems on the same dataset collected from two scenarios. These SOTAs include P-MIMO, which uses RNN to extract multiple received signal strength and DNN for predicting regression location outputs [31]; HDLM, which uses a convolutional neural network (CNN) for RSS feature extractor and LSTM for regression locations estimations [32]; GRU-CNN-net, which uses CNN to extract RSS features and Gated Recurrent Unit (GRU) for positioning predictions [33].

Figure 20 represents the CDF performances of the MM-Loc model and the aforementioned SOTAs models. By observation, within the prediction accuracy of 90%, MM-Loc outperforms all other SOTAs estimation results. The comparison numerical results are shown in table 9. Comparing with other SOTA baseline models, our proposed MM-Loc wins the best performance, which shows the maximum positioning precision with the highest prediction accuracy and the lowest estimation error under both scenarios.

Table 9. Comparative study of model performances under both scenarios.

Method	ScenarioA				ScenarioB			
	Min	Max	Mean	Std	Min	Max	Mean	Std
MM-Loc	**0.0331m**	**30.1591m**	1.5530m	**1.7790m**	**0.0031m**	20.2881m	**1.8859m**	1.9679m
P-MIMO	0.0866m	30.3014m	2.4021m	2.9929m	0.0059m	21.3284m	2.4363m	2.0115m
HDLM	0.0337m	30.9279m	1.9946m	2.5993m	**0.0031m**	**18.4450m**	2.1128m	2.0198m
GRU-CNN-net	0.0348m	33.1648m	**1.5446m**	1.7886m	0.0048m	22.7855m	2.1867m	**1.8698m**

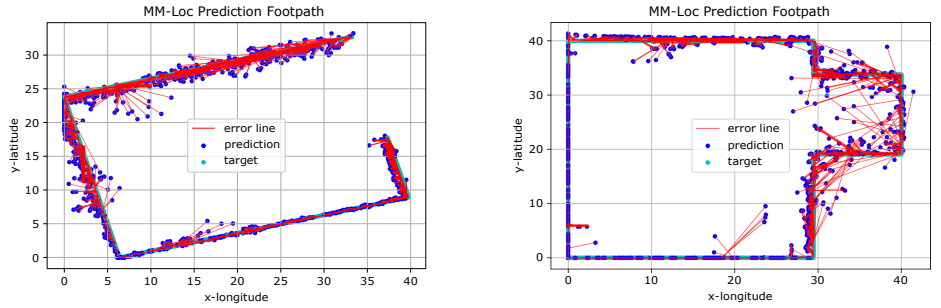


(a) SOTA Performance CDF on Scenario A (b) SOTA Performance CDF on Scenario B

Figure 20. Comparison of model performance CDF of MM-Loc and SOTA models.

6.3. MM-Loc Visualisation

Figure 21 presents the predicted footpath from the MM-Loc for two scenarios. The Red line indicates the distance between the coordinates of the ground truth and estimated locations produced by the MM-Loc model.



(a) MM-Loc Predicted Footpath on Scenario A (b) MM-Loc Predicted Footpath on Scenario B

Figure 21. MM-Loc Footpath Visualisation

We observe that MM-Loc predicts the footpath along the corridor with high fidelity, having clear estimation boundaries. However, some predictions are over 5 metres away from the ground truth, especially at the corners of corridors. This is likely an effect of the difficulty of observations in the WiFi component near corners. The other aspect introducing errors is the magnetic interference present in some places on the pathway (elevators and heavy iron materials in building materials).

7. Discussion

This work shows that conventional smartphone indoor localization methods can be modelled by an end-to-end multimodal deep neural network using diverse sensor streams. This starts with individual feature extractors specific to each modality (using RNNs and DNNs) and then fusing these representations for the final inference through joint neural network architecture. With this, we are moving the effort from engineering each modality component (step counting, direction estimation) and other conventional integration methods (particle filters, Kalman filter and graph-based constraints) to a purely data-driven machine learning approach.

Our data-driven fusion approach builds entirely on the quality and volume of data, without engineering preliminary features nor making assumptions about the use of the system. Previous systems fail when porting to new environments because of the built-in assumptions about the scene. Contrastively, our system is generalizable as it requires low deployment costs. This is because the MM-Loc model can be automatically retrained and re-compatible to the new-deployed scenarios by transfer learning, relying

on the updated sampling data with minimum development costs on data collection and labelling.

Based on the merging methodology that fusing divisive modality features extracted by sub-components, the system is potentially extendable to adapt for more variations and additional signal sources (such as light, environmental noise, humidity, air pressure, etc.) for positioning.

There is still space for improvements in the estimation accuracy. Despite our exploration with a low volume of training data, we show that even this limited training set is enough for our end-to-end machine learning multimodal DNN solution. This method moves the effort entirely on the quality of training data. Although data collection is still a hard challenge for now, we believe this is the only way to capture the fine details that are commonly missed by traditional modelling approaches. On the other hand, this information would always be there to observe and train on if larger volumes of data become available. In the future, this data collection can be automated, either by robots roaming the indoor space to update WiFi radio maps or by mass unlabelled data collection from users roaming naturally in the environment. With labelling solutions based on computer vision [25], this data can be used fruitfully.

8. Conclusions

In this work, we introduce an end-to-end hybrid multimodal deep neural network, MM-Loc, for performing the task of smartphone indoor localisation. Our MM-Loc is an entirely data driven approach. We model the conventional methods of indoor localisation, WiFi fingerprinting and Dead Reckoning through neural network structures. These are capable of performing location estimation independently, with a median error of 2.8 metres by the WiFi DNN model and 6.5 metres by the inertial RNN model, respectively. For fusion we developed MM-Loc as a multimodal structure that bridges feature representations from modality-specific models into a more robust location estimation approach. Our MM-Loc achieves a performance of 1.9 metres median error and is easy to deploy by learning directly from data automatically.

References

1. Misra, P.; Enge, P. Global positioning system: Signals, measurements and performance second edition. *Massachusetts: Ganga-Jamuna Press* **2006**.
2. Radu, V.; Marina, M.K. Himloc: Indoor smartphone localization via activity aware pedestrian dead reckoning with selective crowdsourced wifi fingerprinting. Proc. IPIN 2013. IEEE, 2013.
3. Ari, A.; Chintalapudi, K.K.; Padmanabhan, V.N.; Sen, R. Zee: Zero-Effort Crowdsourcing for Indoor Localization. Proc. MobiCom. ACM, 2012.
4. Yang, Z.; Wu, C.; Liu, Y. Locating in fingerprint space: wireless indoor localization with little human intervention. Proceedings of the 18th annual international conference on Mobile computing and networking. ACM, 2012, pp. 269–280.
5. Xiao, Z.; Wen, H.; Markham, A.; Trigoni, N. Robust pedestrian dead reckoning (R-PDR) for arbitrary mobile device placement. Proc. IPIN. IEEE, 2014.
6. Wang, H.; Sen, S.; Elgohary, A.; Farid, M.; Youssef, M.; Choudhury, R.R. No need to war-drive: Unsupervised indoor localization. Proceedings of MobiSys. ACM, 2012.
7. Chen, C.; Wang, B.; Lu, C.X.; Trigoni, N.; Markham, A. A survey on deep learning for localization and mapping: Towards the age of spatial machine intelligence. *arXiv preprint arXiv:2006.12567* **2020**.
8. Radu, V.; Tong, C.; Bhattacharya, S.; Lane, N.; Mascolo, C.; Marina, M.; Kawsar, F. Multimodal Deep Learning for Activity and Context Recognition. *IMWUT* **2018**, *1*.
9. Chen, Y.; Lymberopoulos, D.; Liu, J.; .; Priyantha, B. FM-based indoor localization. Proc. MobiSys 2012. ACM, 2012.
10. Dey, A.; Roy, N.; Xu, W.; Choudhury, R.R.; Nelakuditi, S. AccelPrint: Imperfections of Accelerometers Make Smartphones Trackable. NDSS, 2014.
11. Marquez, L.; Salgado, J.G. Machine learning and natural language processing **2000**.
12. Jordan, M.I.; Mitchell, T.M. Machine learning: Trends, perspectives, and prospects. *Science* **2015**, *349*, 255–260.
13. Harle, R. A survey of indoor inertial positioning systems for pedestrians. *Communications Surveys and Tutorials* **2013**, *15*.
14. Roy, N.; Wang, H.; Choudhury, R.R. I am a Smartphone and I can Tell my User's Walking Direction. MobiSys. ACM, 2014.
15. Brajdic, A.; Harle, R. Walk Detection and Step Counting on Unconstrained Smartphones. Proc. UbiComp. ACM, 2013.
16. Alzantot, M.; Youssef, M. UPTIME: Ubiquitous Pedestrian Tracking using Mobile Phones. Proc. WCNC. IEEE, 2012.

17. Ferris, B.; Hahnel, D.; Fox, D. Gaussian Processes for Signal Strength-Based Location Estimation. Proc. Robotics Science and Systems. IEEE, 2006.
18. Youssef, M.; Agrawala, A. The Horus location determination system. *Wireless Networks* **2008**, *14*, 357–374.
19. Dai, H.; hao Ying, W.; Xu, J. Multi-layer neural network for received signal strength-based indoor localisation. *Communications* **2016**, *10*.
20. Dayekh, S.; Affes, S.; Kandil, N.; Nerguizian, C. Cooperative Localization in Mines Using Fingerprinting and Neural Networks. Proc. WCNC. IEEE, 2010.
21. Chintalapudi, K.; Padmanabha Iyer, A.; Padmanabhan, V.N. Indoor localization without the pain. Proceedings of the sixteenth annual international conference on Mobile computing and networking. ACM, 2010, pp. 173–184.
22. Wang, X.; Gao, L.; Mao, S.; Pandey, S. CSI-Based Fingerprinting for Indoor Localization: A Deep Learning Approach. *Transactions on Vehicular Technology* **2017**, *66*.
23. Xiao, Z.; Wen, H.; Markham, A.; Trigoni, N. Lightweight map matching for indoor localisation using conditional random fields. Proc. IPSN. IEEE, 2014.
24. Wu, C.; Yang, Z.; Liu, Y.; Xi, W. WILL: Wireless indoor localization without site survey. *IEEE Transactions on Parallel and Distributed Systems* **2013**, *24*, 839–848.
25. Cosma, A.; Radoi, I.E.; Radu, V. CamLoc: Pedestrian Location Detection from Pose Estimation on Resource-constrained Smart Cameras. arXiv:1812.11209. preprint, 2018.
26. Lipton, Z.C.; Berkowitz, J.; Elkan, C. A critical review of recurrent neural networks for sequence learning. *arXiv preprint arXiv:1506.00019* **2015**.
27. Mikolov, T.; Karafiát, M.; Burget, L.; Černockỳ, J.; Khudanpur, S. Recurrent neural network based language model. Eleventh Annual Conference of the International Speech Communication Association, 2010.
28. Wei, X.; Radu, V. Calibrating Recurrent Neural Networks on Smartphone Inertial Sensors for Location Tracking. 2019 International Conference on Indoor Positioning and Indoor Navigation (IPIN). IEEE, 2019, pp. 1–8.
29. Hochreiter, S.; Schmidhuber, J. Long short-term memory. *Neural computation* **1997**, *9*, 1735–1780.
30. Gers, F.A.; Schmidhuber, J.; Cummins, F. Learning to forget: Continual prediction with LSTM **1999**.
31. Hoang, M.T.; Yuen, B.; Dong, X.; Lu, T.; Westendorp, R.; Reddy, K. Recurrent neural networks for accurate RSSI indoor localization. *IEEE Internet of Things Journal* **2019**, *6*, 10639–10651.
32. Poulose, A.; Han, D.S. Hybrid deep learning model based indoor positioning using Wi-Fi RSSI heat maps for autonomous applications. *Electronics* **2021**, *10*, 2.
33. Kang, J.; Lee, J.; Eom, D.S. Smartphone-based traveled distance estimation using individual walking patterns for indoor localization. *Sensors* **2018**, *18*, 3149.