

Convolutional Neural Networks for Efficient Indoor Navigation with Smartphones



Saideep Tiku, Ayush Mittal, and Sudeep Pasricha

1 Introduction

Contemporary outdoor location-based services have transformed how people navigate, travel, and interact with their surroundings. Emerging indoor localization techniques have the potential to extend this outdoor experience across indoor locales. Beyond academics, many privately funded providers in the industry are focusing on indoor location-based services to improve customer experience. For instance, Google can suggest products to its users through targeted indoor location-based advertisements [1]. Stores such as Target in the United States are beginning to provide indoor localization solutions to help customers locate products in a store and find their way to these products [2]. Services provided by these companies combine GPS, cell towers, and Wi-Fi data to estimate the user's location. Unfortunately, in the indoor environment where GPS signals cannot penetrate building walls, the accuracy of these geo-location services can be in the range of tens of meters, which is insufficient in many cases [3].

Radio signals such as Bluetooth, ultra-wideband (UWB) [4], and radio frequency identification (RFID) [5, 6] are commonly employed for the purpose of indoor localization. The key idea is to use qualitative characteristics of radio signals (e.g., signal strength or triangulation) to estimate user location relative to a radio beacon (wireless access point). These approaches suffer from multipath effects, signal attenuation, and noise-induced interference [8]. Also, as these techniques require specialized wireless radio beacons to be installed in indoor locales, they are costly and thus lack scalability for wide-scale deployment.

S. Tiku (✉) · A. Mittal · S. Pasricha
Department of Electrical and Computer Engineering, Colorado State University, Fort Collins, CO, USA
e-mail: saideep@colostate.edu; sudeep@colostate.edu

Wi-Fi-based fingerprinting is perhaps the most popular radio-signal-based indoor localization technique being explored today. Wi-Fi is an ideal radio signal source for indoor localization as most public or private buildings are pre-equipped with Wi-Fi access points (APs). Lightweight middleware-based fingerprinting frameworks have been shown to run in the background to deliver location-based updates on smartphones [29, 37]. Fingerprinting with Wi-Fi works by first recording the strength of Wi-Fi radio signals in an indoor environment at different locations. Then, a user with a smartphone can capture Wi-Fi received signal strength indication (RSSI) data in real-time and compare it to previously recorded (stored) values to estimate their location in that environment. Fingerprinting techniques can deliver an accuracy of 6–8 m [28], with accuracy improving as the density of APs increases. However, in many indoor environments, noise and interference in the wireless spectrum (e.g., due to other electronic equipment, movement of people, and operating machinery) can reduce this accuracy. Combining fingerprinting-based frameworks with dead reckoning can improve this accuracy somewhat [8]. Dead reckoning refers to a class of techniques where inertial sensor data (e.g., from accelerometer and gyroscope) is used along with the previously known position data to determine the current location. Unfortunately, dead reckoning is infamously known to suffer from error accumulation (in inertial sensors) over time. Also, these techniques are not effective for people using wheelchairs or moving walkways.

The intelligent application of machine learning (ML) techniques can help to overcome noise and uncertainty during fingerprinting-based localization [8]. While traditional ML techniques work well at approximating simpler input–output functions, computationally intensive deep learning models are capable of dealing with more complex input–output mappings and can deliver better accuracy. Middleware-based offloading [30] and energy enhancement frameworks [31, 32, 37] may be a route to explore for computation and energy-intensive indoor localization services on smartphones. Furthermore, with the increase in the available computational power on mobile devices, it is now possible to deploy deep learning techniques such as convolutional neural networks (CNNs) on smartphones.

These are a special form of deep neural networks (DNNs) that are purposely designed to for image-based input data. CNNs are well known to automatically identify high-level features in the input images that have the heaviest impact on the final output. This process is known as feature learning. Prior to deep learning, feature learning was an expensive and time-intensive process that had to be conducted manually. CNN has been extremely successful in complex image classification problems and is finding applications in many emerging domains, e.g., self-driving cars [27].

In this chapter, we discuss an efficient framework that uses CNN-based Wi-Fi fingerprinting to deliver a superior level of indoor localization accuracy to a user with a smartphone. Our approach utilizes widely available Wi-Fi APs without requiring any customized/expensive infrastructure deployments. The framework works on a user's smartphone, within the computational capabilities of the device, and utilizes the radio interfaces for efficient fingerprinting-based localization. The main novel contributions of this chapter can be summarized as follows:

- We discuss a newly developed technique to extract images out of location fingerprints, which are then used to train a CNN that is designed to improve indoor localization robustness and accuracy.
- We implemented a hierarchical architecture to scale the CNN, so that our framework can be used in the real world where buildings can have large numbers of floors and corridors.
- We performed extensive testing of our algorithms with the state of the art across different buildings and indoor paths, to demonstrate the effectiveness of our proposed framework.

2 Related Works

Various efforts have been made to overcome the limitations associated with indoor localization. In this section, we summarize a few crucial efforts toward the same.

Numerous RFID-based [5, 6] indoor localization solutions that use proximity-based estimation techniques have been proposed. But the hardware expenses of these efforts increase dramatically with increasing accuracy requirements. Also, these approaches cannot be used with smartphones and require the use of specialized hardware. Indoor localization systems that use UWB [4] and ultrasound [10] have similar requirements for additional (costly) infrastructure and a lack of compatibility for use with commodity smartphones.

Triangulation-based methods, such as [11], use multiple antennas to locate a person or object. But these techniques require several antennas and regular upkeep of the associated hardware. Most techniques therefore favor using the more lightweight fingerprinting approach, often with Wi-Fi signals. UJIIndoorLoc [7] describes a technique to create a Wi-Fi fingerprint database and employs a k-nearest neighbor (KNN)-based model to predict location. Their average accuracy using KNN is 7.9 m. Given the current position (using fingerprinting) of a user walking in the indoor environment, pedestrian dead reckoning can be used to track the user's movement using a combination of microelectromechanical systems (MEMs)-based motion sensors ubiquitously found within contemporary smartphones and other wearable electronics. Dead reckoning techniques use the accelerometer to estimate the number of steps, a gyroscope for orientation, and a magnetometer to determine the heading direction. Such techniques have been employed in [12, 26] but have shown to deliver poor localization accuracy results when used alone.

Radar [12] and IndoorAtlas [26] proposed using hybrid indoor localization techniques. Radar [12] combines inertial sensors (dead reckoning) with Wi-Fi signal propagation models, whereas Indoor Atlas [26] combines information from several sensors such as magnetic, inertial, and camera sensors, for localization. LearnLoc [8] shallow feed-forward neural network models, dead reckoning techniques, and Wi-Fi fingerprinting to trade-off indoor localization accuracy and energy efficiency during localization on smartphones. Similar to LearnLoc, more recent works focus on optimizing and adapting light-weight machine learning techniques for the

purpose of fingerprinting-based indoor localization [28, 34–36]. However, all such techniques are limited by their ability to identify and match complex pattern within RSSI fingerprints. Additionally, a considerable amount of effort needs to be placed into the preprocessing, feature selection, and the tuning of the underlying model. Given these challenges, there is need of robust methodologies and algorithms for the purpose of fingerprinting-based indoor localization.

A few efforts have started to consider deep learning to assist with indoor localization. The work in [13] presents an approach that uses DNNs with Wi-Fi fingerprinting. The accuracy of the DNN is improved by using a hidden Markov model (HMM). The HMM takes temporal coherence into account and maintains a smooth transition between adjacent locations. But our analysis shows that the fine location prediction with the HMM fails in cases such as when moving back on the same path or taking a sharp turn. HMM predictions are also based on the previous position acquired through the DNN and, hence, can be prone to error accumulation. DeepFi [14] and ConFi [15] propose approaches that use the channel state information (CSI) of Wi-Fi signals to create fingerprints. But the CSI information in these approaches was obtained through the use of specialized hardware attached to a laptop. None of the mobile devices available today have the ability to capture CSI data. Due to this limitation, it is not feasible to implement these techniques on smartphones. Deep belief networks (DBNs) [16] have also been used for indoor localization, but the technology is based on custom UWB beacons that lead to very high implementation cost.

In summary, most works discussed so far either have specialized hardware requirements or are not designed to work on smartphones. Also, our real-world implementation and analysis concluded that the abovementioned frameworks slow down as they become resource intensive when being scaled to cover large buildings with multiple floors and corridors.

The framework discussed in this chapter, *CNNLOC*, overcomes the shortcomings of these state-of-the-art indoor localization approaches and was first presented in [33]. *CNNLOC* creates input images by using RSSI of Wi-Fi signals that are then used to train a CNN model, without requiring any specialized hardware/infrastructure. *CNNLOC* is easily deployable on current smartphones. The proposed framework also integrates a hierarchical scheme to enable scalability for large buildings with multiple floors and corridors/aisles.

3 Convolutional Neural Networks

Convolutional neural networks (CNNs) are specialized form of neural networks (NNs) that are designed for the explicit purpose of image classification [9]. They are highly resilient to noise in the input data and have shown to deliver excellent results for complex image classification tasks. The smallest unit of any neural network is

a perceptron and is inspired by the biological neuron present in the human brain. A perceptron is defined by the following equation:

$$y = \sum_{i=1}^n w_i x_i + w_0 \quad (1)$$

Here, y is the output, which is a weighted sum of the inputs x_i , with a bias (w_0). NNs have interconnected layers, and in each layer, there are several perceptrons, each with its own tunable weights and biases. Each layer receives some input, executes a dot product, and passes it to the output layer or the hidden layer in front of it [17]. An activation function is applied to the output y , limiting the range of values that it can take and establishes an input–output mapping defined by logistic regression. The most common activation functions used are *sigmoid* and *tanh* functions. The goal of an NN is to approximate a functional relationship between a set of inputs and outputs (training phase). The resulting NN then represents the approximated function that is used to make predictions for any given input (testing phase).

While an NN often contains a small number of hidden layers sandwiched between the input and output layer, a deep neural network (DNN) has a very large number of hidden layers. DNNs have a much higher computational complexity but in turn are also able to deliver very high accuracy. CNNs are a type of DNN that include several specialized NN layers, where each layer may serve a unique function. CNN classifiers are used to map input data to a finite set of output classes. For instance, given different animal pictures, a CNN model can be trained to categorize them into different classes such as cats and dogs. CNNs also make use of rectified linear units (ReLUs) as their activation function, which allows them to handle nonlinearity in the data.

In the training phase, our CNN model uses a stochastic gradient descent (SGD) algorithm. Adam [18] is an optimized variant of SGD and is used to optimize the learning process. The algorithm is designed to take advantage of two well-known techniques: RMSprop [19] and AdaGrad [20]. SGD maintains a constant learning rate for every weight update in the network. In contrast, Adam employs an adaptive learning rate for each network weight, with the learning rate being adapted as the training progresses. RMSprop uses the mean (first-order moment) of past-squared gradients and adjusts the weights based on how fast the gradient changes. Adam, to optimize the process, uses the variance (second-order moment) of past gradients and adjusts the weights accordingly.

The structure of the CNN in *CNNLOC* is inspired from the well-known CNN architectures, LeNet [21] and AlexNet [22]. Our CNN architecture is shown in Fig. 1. For the initial set of layers, our model has 2-D convolutional layer, followed by dense layers and culminates in an output layer. The 2-D convolutional layer works by convolving a specific region of the input image at a time. This region is known as a filter. The filter is shown by a rectangle (red-dotted lines). Each layer performs a convolution of a small region of the input image with the filter and feeds the result to

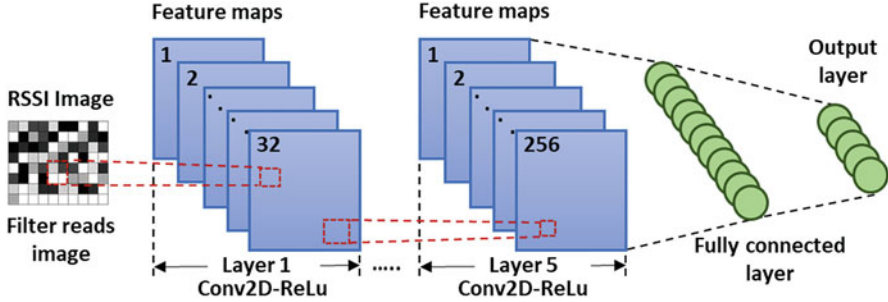


Fig. 1 CNN architecture

the ReLu activation function. Therefore, we refer to each layer as [Conv2D-ReLu]. To capture more details from the input image we can use a larger number of filters. For each filter, we get a feature map. For the first layer of [Conv2D-ReLU], we used 32 filters to create a set of 32 feature maps. We used five hidden layers of [Conv2D-ReLU], but only two are shown for brevity. The number of filters and layers is derived through empirical analysis as discussed in Sect. 4.4. A “stride” parameter determines the quantity of pixels that a filter will shift, to arrive at a new region of the input image to process. The stride and other “hyperparameters” of our CNN are further discussed in Sect. 4.4. In the end, a fully connected layer helps in identifying the individual class scores (in our case each class is a unique location). The class with the highest score is selected as the output. In this layer, all the neurons are connected to the neurons in the previous layer (green-dotted lines).

In a conventional CNN, a pooling layer is used to down-sample the image when the size of the input image is too big. In our case, the input image is small, and therefore, we do not need this step. We want our CNN to learn all the features from the entire image.

4 CNNLOC Framework: Overview

4.1 Overview

An overview of our *CNNLOC* indoor localization framework is shown in Fig. 2. In the framework, we utilize the available Wi-Fi access points (APs) in an indoor environment to create an RSSI fingerprint database. Our framework is divided into two phases. The first phase involves RSSI data collection, cleaning, and preprocessing. This preprocessed data is used to create a database of images. Each image represents a Wi-Fi RSSI-based signature that is unique to a location label. Each location label is further associated with an x - y coordinate. This database of images is used to train a CNN model. The trained model is deployed on to a smartphone. In the second phase, or the online phase, real-time AP data is converted

into an image and then fed to the trained CNN model to predict the location of the user. The CNN model predicts the closest block that was sampled as the users' location. A detailed description of the preprocessing is described in the next section.

4.2 Preprocessing of RSSI Data

The process of image database creation begins with the collection of RSSI fingerprints as shown in the top half of Fig. 2. The RSSI for various APs are captured along with the corresponding location labels and x - y coordinates. Each AP is uniquely identified using its unique media access control (MAC) address. We only maintain information for known Wi-Fi APs and hence clean the captured data. This ensures that our trained model is not polluted by unstable Wi-Fi APs. On the RSSI scale, values typically range between -99 dB (lowest) and -0 dB (highest). To indicate that a specific AP is unreachable, -100 is used, or no signal is received from it. We normalize the RSSI values on a scale from 0 and 1, where 0 represents no signal and 1 represents the strongest signal.

Assume that while fingerprinting an indoor location, a total of K APs are discovered at N unique locations. These combine to form a two-dimensional matrix of size $N \times K$. Then, the normalized RSSI fingerprint at the N th location, denoted as l_N , is given by a row vector $[r_1, r_2, \dots, r_K]$, denoted by R_N . Therefore, each column vector, $[w_1, w_2, \dots, w_N]$ would represent the normalized RSSI values of the K th AP at all N locations, denoted by W_K . We calculate the Pearson correlation coefficient (PCC) [23] between each column vector W_K and the location vector $[l_1, l_2, \dots, l_N]$. The result is a vector of correlation values denoted as C . PCC is useful in identifying the most significant APs in the database that impact localization

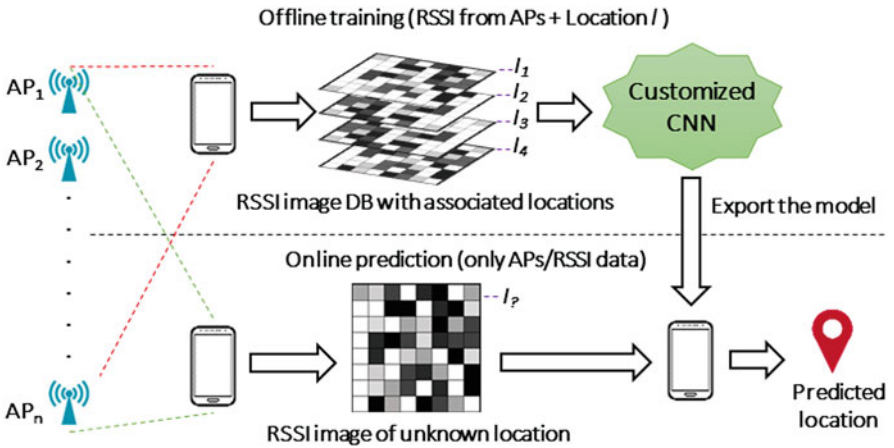


Fig. 2 An overview of the CNNLOC framework

accuracy. The coefficient values range across a scale of -1 to $+1$. If the relationship is -1 , it represents a strong negative relationship, whereas $+1$ represents a strong positive relationship, and 0 implies that the input and output have no relationship.

We only consider the magnitude of the correlation as we are only concerned with the strength of the relationship. APs with very low correlation with the output coordinates are not useful for the purpose of indoor localization. Therefore, we can remove APs whose correlation to the output coordinates is below a certain threshold ($|PCC| < 0.3$). This removes inconsequential APs from the collected Wi-Fi data and helps reduce the computational workload of the framework. The normalized RSSI data from the remaining high-correlation APs is used to create an RSSI image database, as explained in the next section.

4.3 RSSI Image Database

In this section, we present our approach to convert RSSI data for a given location into a gray scale image. A collection of these images for all fingerprinted locations forms the RSSI image database. To form gray scale images, a Hadamard product (*HP*) [24] is calculated for each R and C . *HP* is defined as an element wise multiplication of two arrays or vectors:

$$HP = \sum_{i=1}^N R_i \circ C \quad (2)$$

The dimension of each *HP* is $1 \times K$. Then, the *HP* matrix is reshaped into a $p \times p$ matrix, which represents a 2-D image as shown in Fig. 3. The *HP* is padded with zeros in the case that K is less than p^2 . Therefore, we now have a set of N images of size $p \times p$ in our database. These images are used to train the CNNs.

Figure 3 shows two images of size 8×8 created for two unique fingerprints (signatures) associated with two different locations. Each pixel value is scaled on a scale of $0-1$. The patterns in each of these images will be unique to a location and change slightly as we move along an indoor path.

In Eq. (2), the product of PCC and normalized RSSI value for each AP is used to form a matrix. Its purpose is to promote the impact of the APs that are highly correlated to fingerprinted locations. Even though there may be attenuation of Wi-Fi signals due to multipath fading effects, the image may fade but will likely still have the pattern information retained. These patterns that are unique to every location can be easily learned by a CNN. The hyperparameters and their use in *CNNLOC* are discussed next.

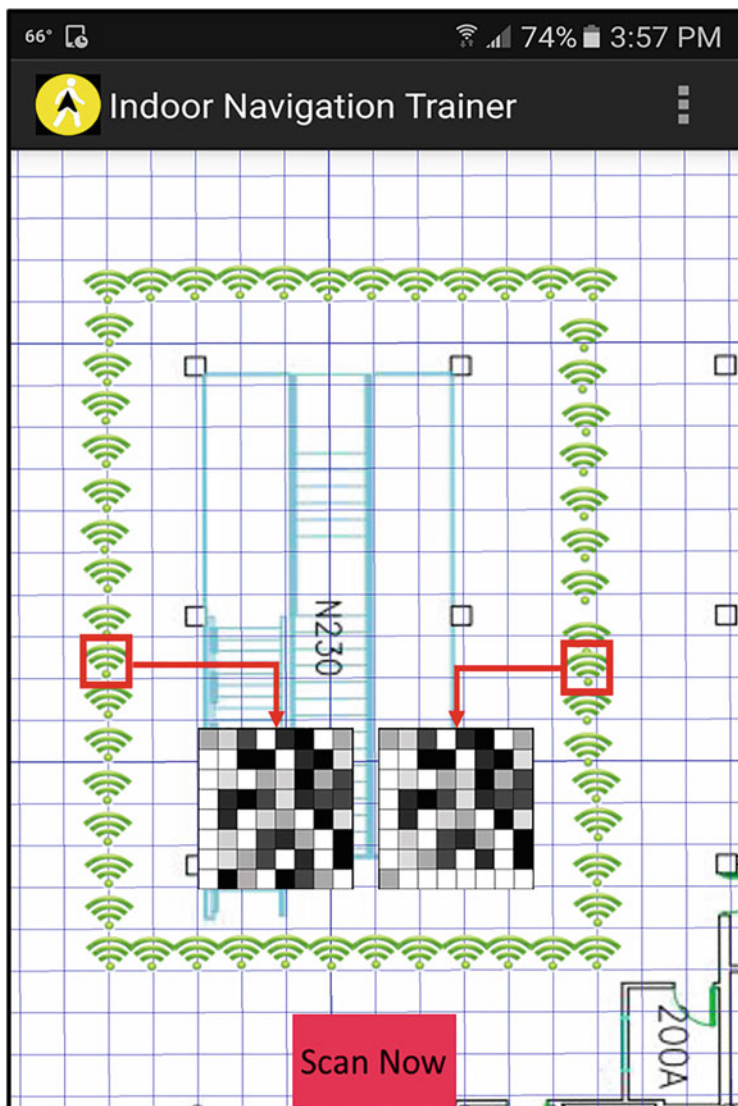


Fig. 3 Snapshot of CNNLOC’s offline phase application showing contrasting the images created for two unique locations. The green icons represent locations that are fingerprinted along an indoor path. The two locations shown are 10 m apart

4.4 Hyperparameters

The accuracy of the CNN model depends on the optimization of the hyperparameters that control its architecture, which is the most important factor in the

performance of CNN. A smaller network may not perform well, and a larger network may be slow and prone to overfitting. There are no defined rules in deep learning that help in estimating the appropriate hyperparameters and therefore need to be empirically found through an iterative process. The estimated hyperparameters are also highly dependent on the input dataset. For the sake of repeatability, we discuss some the key hyperparameters of our CNN model below:

- *Number of hidden layers*: A large number of hidden layers lead to longer execution times and conversely, fewer hidden layers may produce inaccurate results due to the challenges associated with vanishing gradients. We found that five layers of [Conv2D-ReLU] worked best for our purposes.
- *Size of filter*: This defines the image area that the filter considers at a time, before moving to the next region of the image. A large filter size might aggregate a large chunk of information in one pass. The optimum filter size in our case was found to be 2×2 .
- *Stride size*: The number of pixels a filter moves by is dictated by the stride size. We set it to 1 because the size of our image is very small, and we do not wish to lose any information.
- *Number of filters*: Each filter extracts a distinct set of features from the input to construct different feature maps. Each feature map holds unique information about the input image. The best results were obtained if we started with a lower number of filters and increased them in the successive layers to capture greater uniqueness in the patterns. There were 32 filters in the first layer and were doubled for each subsequent layer up to 256 filters such that both the fourth and fifth layer had 256 filters.

4.5 Integrating Hierarchy for Scalability

We architect our *CNNLOC* framework to scale up to larger problem sizes than that handled by most prior efforts. Toward this, we enhanced our framework by integrating a hierarchical classifier. The resulting hierarchical classifier employs a combination of smaller CNN modules, which work together to deliver a location prediction. Figure 4 shows the hierarchical classification structure of the framework. Each CNN model has a label that starts with C. The C1 model classifies the floor numbers, and then in the next layer, C20 or C21 identifies the corridor on that floor. Once the corridor is located, one of the CNNs from the third layer (C30–C35) will predict the fine-grain location of the user. This hierarchical approach can further be extended across buildings.

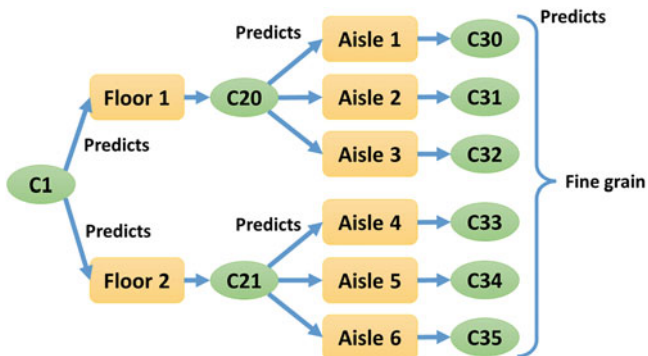


Fig. 4 A general architecture for the hierarchical classifier

Table 1 Indoor paths used in experiments

| Path name | Length (m) | Shape |
|-----------|------------|----------------|
| Library | 30 | U shape |
| Clark A | 35 | Semi-octagonal |
| Physics | 28 | Square shape |

5 Experiments

5.1 Experimental Setup

The following sections describe the *CNNLOC* implementation and experimental results that were conducted on three independent indoor paths as described in Table 1. The overall floor plan of the path is divided into a grid and tiles of interest are labeled sequentially from 1 to N . For the purposes of this work, each square in the grid has an area of 1 m^2 . Based on our analysis (not presented here), having grid tiles of size smaller than 1 m^2 did not lead to any improvements. Each of these labeled tiles is then treated as a “class.” This allows us to formulate indoor localization as a classification problem. Figure 5 shows an example of a path covered in the library building floor plan with labeled squares. Each label further translates into an x - y coordinate. Five Wi-Fi scans were conducted at each square during the fingerprinting (training) phase.

5.2 Smartphone Implementation

An android application was built to collect Wi-Fi fingerprints (i.e., RSSI samples from multiple APs at each location) and for testing. The application is compatible with Android 6.0 and was tested on a Samsung Galaxy S6. After fingerprint data collection, the data was preprocessed as described in the previous section for the CNN model. The entire dataset is split into training and testing samples, so we can

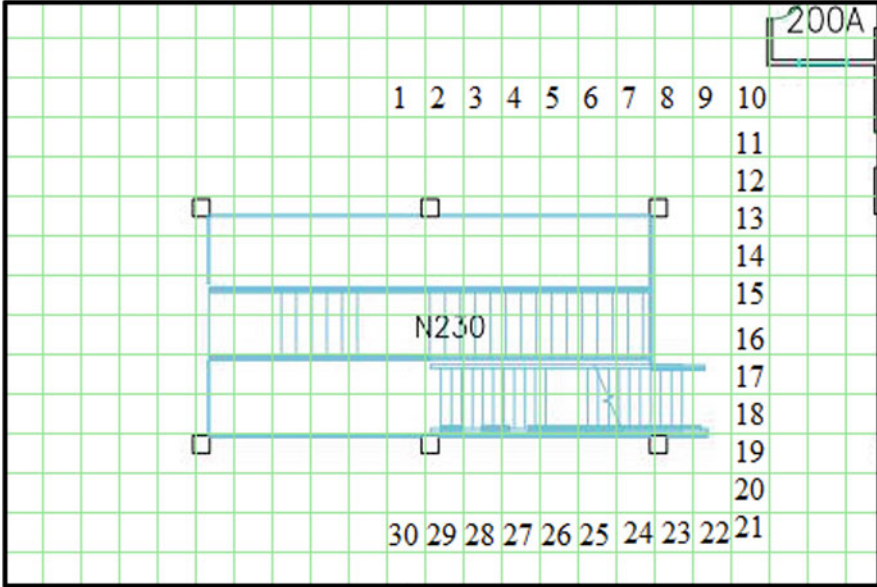


Fig. 5 Library building path divided into a grid, with squares along the path labeled sequentially from 1 to 30

check how well our models perform. We used one-fifth of the total samples for testing, and four-fifth of the samples were used for training.

5.3 *Experimental Results*

We compared our *CNNLOC* indoor localization framework with three other indoor localization frameworks from prior work. The first work we implemented is based on the approach in [25] and employs support vector regression (SVR). The approach forms one or more hyperplanes in a multidimensional space segregating similar data point, which are then used for regression. The second work is based on the KNN technique from [8], which is a nonparametric approach that is based on the idea that similar input will have similar outputs. Lastly, we compare our work against a DNN based approach [13] that improves upon conventional NNs by incorporating a large number of hidden layers. All of these techniques supplement the Wi-Fi fingerprinting approach with a machine learning model to provide robustness against noise and interference effects. Our experiments in the rest of this section first discuss the localization accuracy results for the techniques. Subsequently, we also discuss results for the scalability of our framework using a hierarchical classification enhancement approach. Lastly, we contrast the accuracy of our framework with that reported by other indoor localization techniques.

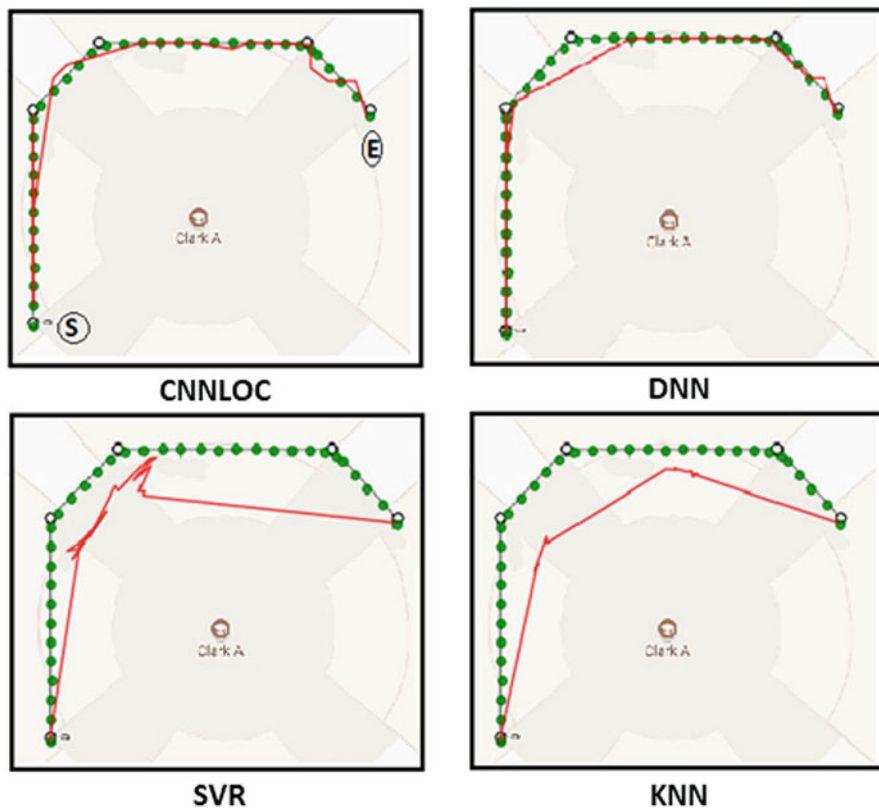


Fig. 6 Path traced using different techniques at the Clark building path. Green and red traces indicate actual and predicted paths respectively

5.3.1 Indoor Localization Accuracy Comparison

The overall indoor localization quality as experienced by a user is heavily impacted by the stability of the predicted path that is traced over an indoor localization session. In an attempt to evaluate this, we compare the paths traced by various indoor localization frameworks as compared with the proposed *CNNLOC* framework. Figure 6 shows the paths predicted by the four techniques, for the indoor path in the Clark building. The green dots along the path represent the points where Wi-Fi RSSI fingerprint samples were collected to create the training fingerprint dataset. The distance between each of the green dots is 1 m. In the offline phase, the RSSI fingerprint at each green dot is converted into an image. The online phase consists of the user walking along this path, and the red lines in Fig. 6 represent the paths predicted by the four techniques. It is observed that KNN [8] and SVR [25] stray off the actual path the most, whereas DNN and *CNNLOC* perform much better. This is likely because KNN and SVR are both regression-based techniques where

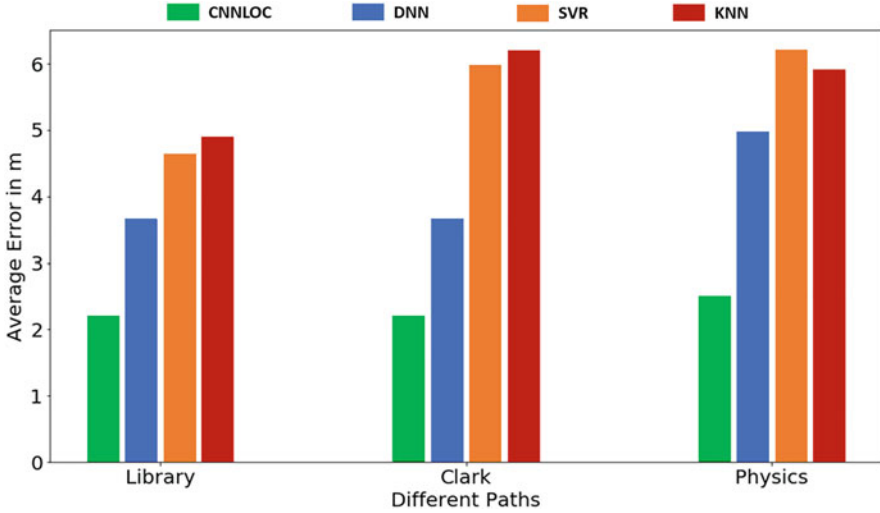


Fig. 7 Comparison of indoor localization techniques

the prediction is impacted by neighboring data points in the RSSI Euclidean space. Two locations that have RSSI fingerprints that are very close to each other in the Euclidian space might not be close to each other on the actual floor plan. This leads to large localization errors, especially when utilizing regression-based approaches. The transition from one location to another is smoother for CNN as it is able to distinguish between closely spaced sampling locations due to our RSSI-to-image conversion technique. The convolutional model is able to identify patterns within individual RSSI images and classify them as locations. From Fig. 6, it is evident that our *CNNLOC* framework produces stable predictions for the Clark path.

Figure 7 shows a bar graph that summarizes the average location estimation error in meters for the various frameworks on the three different indoor floor plans considered. We found that the KNN approach is the least reliable among all techniques with a mean error of 5.5 m and large variations across the paths. The SVR-based approach has a similar mean error as the KNN approach. The DNN-based approach shows lower error across all of the paths. But it does not perform consistently across all of the paths, and the mean error is always higher than that for *CNNLOC*. This may be due to the fact that the filters in CNN are set up to focus on the image with a much finer granularity than the DNN approach is capable of. We also observe that all techniques perform the worst in the Physics department. This is due to the fact that the path in the Physics department is near the entrance of the building and has a lower density of Wi-Fi APs as compared with the other paths. The Library and Clark paths have a higher density of Wi-Fi APs present; hence, better accuracy can be achieved. Our proposed *CNNLOC* framework is the most reliable framework with the lowest mean error of less than 2 m.

5.3.2 CNNLOC Scalability Analysis

The size and complexity of a deep learning model is directly correlated to number of classes and associated dataset in use. The baseline formulation of our proposed framework does not account for the increasing area of floor plan that needs to be covered. To overcome this, we proposed a hierarchal approach for *CNNLOC* (Sect. 4.5). We consider a scenario when *CNNLOC* is required to predict a location inside a building with two floors and with three corridors on each floor. The length of each corridor is approximately 30 m. We combined several small CNNs (in our case 9 small CNNs), such that a smaller number of weights are associated with each layer in the network than if a single larger CNN was used.

We first evaluated the accuracy of predictions, for *CNNLOC* with and without the hierarchical classifier. For the first and second layer of the hierarchical classifier (shown in Fig. 4), the accuracy is determined by the number of times the system predicts the correct floor and corridor. We found that floors and corridors were accurately predicted 99.67% and 98.36% of times, respectively. For the final layer, we found that there was no difference in accuracy between the hierarchal approach and the nonhierarchal approach. This is because in the last level, both the approaches use the same model.

Figure 8 shows the benefits in terms of time taken to generate a prediction with the hierarchical versus the nonhierarchal *CNNLOC* framework. We performed our experiment for four walking scenarios (“runs”) in the indoor environment (building with two floors and with three corridors on each floor). We found that the hierarchical *CNNLOC* model only takes 2.42 ms to make a prediction on average, whereas the nonhierarchal *CNNLOC* takes longer (3.4 ms). Thus, the proposed hierarchical classifier represents a promising approach to reduce prediction time due to the fewer number of weights in the CNN layers in the hierarchical approach, which leads to fewer computations in real time.

5.3.3 Accuracy Analysis with Other Approaches

Our experimental results in the previous sections have shown that *CNNLOC* delivers better localization accuracy over the KNN [8], DNN [13], and SVR [25] frameworks. The UJIIndoorLoc [7] framework is reported to have an accuracy of 4–7m. Our average accuracy is also almost twice that of RADAR [12]. If we consider frameworks that used CSI (DeepFi [14] and ConFi [15]), our accuracy is very close to both at just under 2 m. However, [14, 15] use special equipment to capture CSI and cannot be used with mobile devices. In contrast, our proposed *CNNLOC* framework is easy to deploy on today’s smartphones, does not require any specialized infrastructure (e.g., custom beacons), and can be used in buildings wherever Wi-Fi infrastructure preexists.

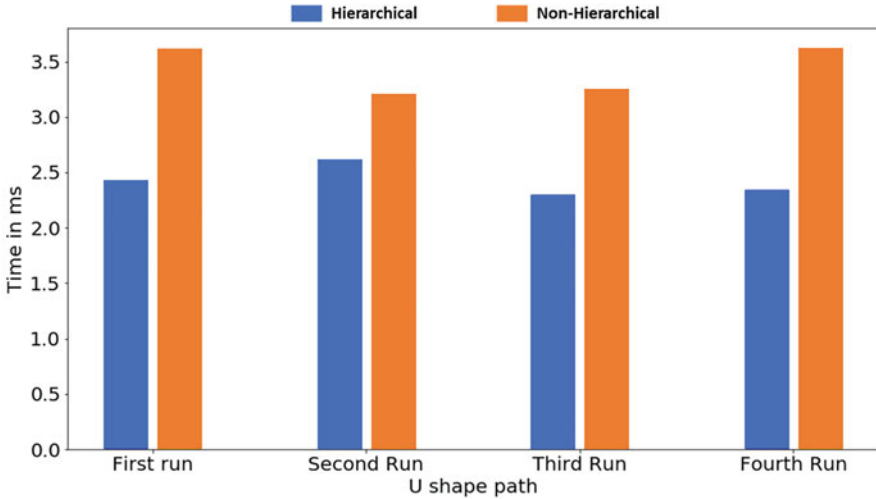


Fig. 8 A comparison of execution times for hierarchical and nonhierarchical versions the CNNLOC framework

6 Conclusion

In this chapter, we discuss the *CNNLOC* framework [33] that uses Wi-Fi fingerprints and convolutional neural networks (CNNs) for accurate and robust indoor localization. We compared our work against three different state-of-the-art indoor localization frameworks from prior work. Our framework outperforms these approaches and delivers localization accuracy under 2 m. *CNNLOC* has the advantage of being easily implemented without the overhead of expensive infrastructure and is smartphone compatible. We also demonstrated how a hierarchical classifier can improve the scalability of this framework. *CNNLOC* represents a promising framework that can deliver reliable and accurate indoor localization for smartphone users.

Acknowledgments This work was supported by the National Science Foundation (NSF), through grant CNS-2132385.

References

1. How Google Maps Makes Money. (2022) [Online] <https://www.investopedia.com/articles/investing/061115/how-does-google-maps-makes-money.asp>. Accessed 1 Apr 2022
2. Target Rolls Out Bluetooth Beacon Technology in Stores to Power New Indoor Maps in its App. (2017) [Online] <https://techcrunch.com/2017/09/20/target-rolls-out-bluetooth-beacon-technology-in-stores-to-power-new-indoor-maps-in-its-app/>. Accessed 1 Apr 2022

3. Case Study: Accuracy & Precision of Google Analytics Geolocation. (2017) [Online] Available at: <https://radical-analytics.com/case-study-accuracy-precision-of-google-analytics-geolocation-4264510612c0>. Accessed 1 Dec 2017
4. Ubisense Research Network. [Online] Available at: <http://www.ubisense.net/>. Accessed 1 Dec 2017
5. Jin, G., Lu, X., Park, M.: An indoor localization mechanism using active RFID tag. In: IEEE International Conference on Sensor Networks, Ubiquitous, and Trustworthy Computing (SUTC). IEEE (2006)
6. Chen, Z., Wang, C.: Modeling RFID signal distribution based on neural network combined with continuous ant colony optimization. *Neurocomputing*. **123**, 354–361 (2014)
7. Torres-Sospedra, J., et al.: UJIIndoorLoc: a new multi-building and multi-floor database for WLAN fingerprint-based indoor localization problems. In: IEEE Indoor Positioning and Indoor Navigation (IPIN). IEEE (2014)
8. Pasricha, S., Ugave, V., Han, Q., Anderson, C.: LearnLoc: a framework for smart indoor localization with embedded mobile devices. In: ACM/IEEE International Conference on Hardware/Software Codesign and System Synthesis (CODES+ISSS). IEEE (2015)
9. Pouyanfar, S., Sadiq, S., Yan, Y., Tian, H., Tao, Y., Reyes, M.P., Shyu, M., Chen, S., Iyengar, S.S.: A survey on deep learning: algorithms, techniques, and applications. *ACM Comput. Surv.* **51**(5), 1–36 (2019)
10. Borriello, G., Liu, A., Offer, T., Palistrant, C., Sharp, R.: WALRUS: wireless acoustic location with room-level resolution using ultrasound. In: Mobile systems, applications, and services (MobiSys). ACM (2005)
11. Yang, C., Shao, H.R.: WiFi-based indoor positioning. *IEEE Commun. Mag.* **53**(3), 150–157 (2015)
12. Bahl, P., Padmanabhan, V.: RADAR: an in-building RF-based user location and tracking system. In: IEEE International Conference on Computer Communications (INFOCOM). IEEE (2000)
13. Zhang, W., Liu, K., Zhang, W., Zhang, Y., Gu, J.: Deep neural networks for wireless localization in indoor and outdoor environments. *Neurocomputing*. **194**, 279–287 (2016)
14. Wang, X., Gao, L., Mao, S., Pandey, S.: DeepFi: deep learning for indoor fingerprinting using channel state information. In: IEEE Wireless Communications and Networking Conference (WCNC). IEEE (2015)
15. Chen, H., Zhang, Y., Li, W., Tao, X., Zhang, P.: ConFi: convolutional neural networks based indoor WiFi localization using channel state information. *IEEE Access*. **5**, 18066–18074 (2017)
16. Hua, Y., Guo, J., Zhao, H.: Deep belief networks and deep learning. In: IEEE International Conference on Intelligent Computing and Internet of Things (ICIT). IEEE (2015)
17. Stanford CNN Tutorial. [Online] Available at: <http://cs231n.github.io/convolutional-networks>. Accessed 1 Apr 2022
18. Kingma, D.P., Ba, J.: Adam: a method for stochastic optimization. In: International Conference on Learning Representations (ICLR) (2015)
19. RMSProp. [Online] https://www.cs.toronto.edu/~tijmen/csc321/slides/lecture_slides_lec6.pdf. Accessed 1 Apr 2022
20. Duchi, J., Hazan, E., Singer, Y.: Adaptive subgradient methods for online learning and stochastic optimization. *ACM J. Mach. Learn. Res.* **12**, 2121–2159 (2011)
21. Lecun, Y., Bottou, L., Bengio, Y., Haffner, P.: Gradient-based learning applied to document recognition. *Proc. IEEE*. **86**(11), 2278–2324 (1998)
22. Krizhevsky, A., Sutskever, I., Hinton, G.E.: ImageNet classification with deep convolutional neural networks. In: Advances in Neural Information Processing Systems. Neural IPS 2012 (2012)
23. Benesty, J., Chen, J., Huang, Y., Cohen, I.: Pearson correlation coefficient. In: Topics in Signal Processing, vol. 2, pp. 1–4. Springer (2009)
24. Styan, G.: Hadamard products and multivariate statistical analysis. In: Linear Algebra and its Applications, vol. 6, pp. 217–240. Elsevier (1973)

25. Cheng, Y.K., Chou, H.J., Chang, R.Y.: Machine-learning indoor localization with access point selection and signal strength reconstruction. In: IEEE Vehicular Technology Conference (VTC). IEEE (2016)
26. IndoorAtlas. [Online] <http://www.indooratlas.com/>. Accessed 1 Apr 2022
27. Rausch, V., Hansen, A., Solowjow, E., Liu, C., Kreuzer, E., Hedrick, J.K.: Learning a deep neural net policy for end-to-end control of autonomous vehicles. In: IEEE American Control Conference (ACC). IEEE (2017)
28. Langlois, C., Tiku, S., Pasricha, S.: Indoor localization with smartphones: harnessing the sensor suite in your pocket. *IEEE Consum. Electron.* **6**(4), 70–80 (2017)
29. Tiku, S., Pasricha, S.: Energy-efficient and robust middleware prototyping for smart mobile computing. In: IEEE International Symposium on Rapid System Prototyping (RSP). IEEE (2017)
30. Khune, A., Pasricha, S.: Mobile network-aware middleware framework for energy-efficient cloud offloading of smartphone applications. In: IEEE Consumer Electronics. IEEE (2017)
31. Donohoo, B., Ohlsen, C., Pasricha, S.: A middleware framework for application-aware and user-specific energy optimization in smart Mobile devices. *Journal Pervasive Mob. Comput.* **20**, 47–63 (2015)
32. Donohoo, B., Ohlsen, C., Pasricha, S., Anderson, C., Xiang, Y.: Context-aware energy enhancements for smart mobile devices. *IEEE Trans. Mob. Comput.* **13**(8), 1720–1732 (2014)
33. Mittal, A., Tiku, S., Pasricha, S.: Adapting convolutional neural networks for indoor localization with smart mobile devices. In: ACM Great Lakes Symposium on VLSI (GLSVLSI). ACM (2018)
34. Tiku, S., Pasricha, S., Notaros, B., Han, Q.: SHERPA: a lightweight smartphone heterogeneity resilient portable indoor localization framework. In: IEEE International Conference on Embedded Software and Systems (ICCESS). IEEE (2019)
35. Tiku, S., Pasricha, S.: PortLoc: a portable data-driven indoor localization framework for smartphones. *IEEE Des. Test.* **36**(5), 18–26 (2019)
36. Tiku, S., Pasricha, S., Notaros, B., Han, Q.: A hidden markov model based smartphone heterogeneity resilient portable indoor localization framework. *J. Syst. Archit.* **108**, 101806 (2020)
37. Pasricha, S., Ayoub, R., Kishinevsky, M., Mandal, S.K., Ogras, U.Y.: A survey on energy management for mobile and IoT devices. *IEEE Des. Test.* **37**(5), 7–24 (2020)