Adapting Convolutional Neural Networks for Indoor Localization with Smart Mobile Devices

Ayush Mittal, Saideep Tiku, Sudeep Pasricha
Department of Electrical and Computer Engineering
Colorado State University, Fort Collins, CO, U.S.A.
ayusmittal@gmail.com, {saideep, sudeep}@colostate.edu

ABSTRACT

Indoor localization is emerging as an important application domain for enhanced navigation (or tracking) of people and assets in indoor locales such as buildings, malls, and underground mines. Most indoor localization solutions proposed in prior work do not deliver good accuracy without expensive infrastructure (and even then, the results may lack consistency). Ambient wireless received signal strength indication (RSSI) based fingerprinting using smart mobile devices is a low-cost approach to the problem. However, creating an accurate 'fingerprinting-only' solution remains a challenge. This paper presents a novel approach to transform Wi-Fi signatures into images, to create a scalable fingerprinting framework based on Convolutional Neural Networks (CNNs). Our proposed CNN based indoor localization framework (*CNN-LOC*) is validated across several indoor environments and shows improvements over the best known prior works, with an average localization error of < 2 meters.

KEYWORDS

Indoor localization; convolutional neural networks, (CNN), mobile devices; Wi-Fi fingerprinting; deep learning

ACM Reference Format:
Ayush Mittal, Saideep Tiku, Sudeep Pasricha. 2018. Adapting Convolutional Neural Networks for Indoor Localization with Smart Mobile Devices. In GLSVLSI '18: 2018 Great Lakes Symposium on VLSI, May 23–25, 2018, Chicago, IL, USA. ACM, NY, NY, USA, 6 pages.
http://dx.doi.org/10.1145/3194554.3194594

1. INTRODUCTION

Existing outdoor location based services have transformed how people navigate, travel, and interact with the world around them. Now, indoor localization techniques are emerging that have the potential to extend this outdoor experience across indoor locales. Industry is beginning to provide 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 USA 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. However, 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].

GLSVLSI '18, May 23–25, 2018, Chicago, IL, USA

ACM ISBN 978-1-4503-5724-1/18/05…$15.00
https://doi.org/10.1145/3194554.3194594

Many of the latest indoor localization techniques exploit radio signals, such as Bluetooth, UWB (Ultra-Wide Band) [4], RFID (Radio Frequency Identification) [5], [6], or other customized radios. The key idea is to use characteristics of radio signals (e.g., signal strength or triangulation) to estimate user location relative to a radio beacon (wireless access point). But these techniques 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.

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]. 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 to 8 meters [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, operating machinery, etc.) 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, gyroscope) is used along with the previously known position data to determine the current location. But dead reckoning is 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 use 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] 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. A CNN is a special type of Deep Neural Network (DNN) that is geared towards image matching and recognition. The most popular aspect of CNN is that it can automatically identify essential input features that make the most impact towards the correctness of the final output. This process is known as feature learning. Prior to deep learning, feature learning was an ex-

pensive 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 paper, we propose a new and 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 paper can be summarized as follows:

- We developed a new technique to extract images out of location fingerprints, which are then used to train a CNN 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 WORK

Several efforts aim to address the challenges in the domain of indoor localization. Here we summarize some of the key efforts.

Several RFID [5], [6] based 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 KNN (K-Nearest Neighbor) based model to predict location. Their average accuracy using KNN is 7.9 meters. 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] and [26], but have shown to deliver poor localization accuracy results when used alone.

Radar [12] and Indoor Atlas [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] combines non-deep ML models, dead reckoning techniques, and Wi-Fi fingerprinting to trade-off indoor localization accuracy and energy efficiency during localization on smartphones.

A few efforts have begun 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 (DBN) [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 of the above-mentioned frameworks either require additional costly infrastructure or cannot be deployed on smart mobile devices. Our implementation-based analysis shows that these frameworks can become slow and resource intensive if used for large buildings with multiple floors and corridors.

Our proposed framework in this paper, *CNN-LOC*, overcomes the shortcomings of these state-of-the-art indoor localization approaches. *CNN-LOC* 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. *CNN-LOC* is easily deployable on current smartphones. The 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 DNNs with a focus on image classification. 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 (NN) 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 weighted bias (w_0). NNs have inter-connected 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]. This output is often applied to an activation function that gives 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, dogs, etc. CNNs also make use of Rectified Linear Units (ReLUs) as their activation function, which allows them to handle non-linearity in the data.

In the training phase, our CNN model uses a feed forward deep

learning algorithm. To update the weights during the training phase, a Stochastic Gradient Descent (SGD) algorithm is used. Adam [18], an optimized version of SGD, 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.

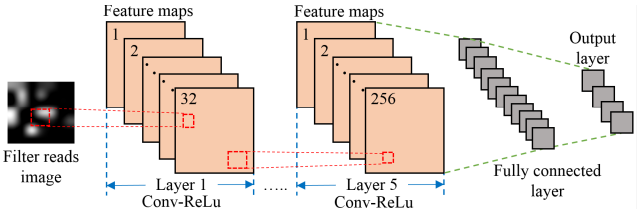

Figure 1: CNN architecture

The structure of the CNN in *CNN-LOC* is inspired from the well-known CNN architectures, LeNet [21] and AlexNet [22]. Our CNN architecture is shown in figure 1. The first hidden layer is partially connected to the input layer. This hidden layer only looks at a specific region of the input image at a time, and 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 the ReLu activation function. Therefore, we refer to each layer as [Conv-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 [Conv-ReLU], we used 32 filters to create a set of 32 feature maps. We used five hidden layers of [Conv-ReLU], but only two are shown for brevity. The number of filters and layers are derived through empirical analysis as discussed in section 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 section 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. *CNN-LOC* FRAMEWORK: OVERVIEW

4.1 Overview

An overview of our *CNN-LOC* indoor localization framework is shown in figure 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 pre-processing. This pre-processed data is used to create a database of images. Each image represents a Wi-Fi RSSI based signature that is unique to a location (i.e., x-y co-ordinate). This database of images is used to train a CNN model. The trained model is deployed on a smartphone. In the second 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 pre-processing is described in the next section.

4.2 Pre-processing of RSSI Data

The process of image database creation begins with the collection of RSSI fingerprints as shown in the top half of figure 2. The RSSI for various APs are captured along with the corresponding x and y coordinates at the training locations. 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 -95 dB (lowest) to -0 dB (highest). We normalize the RSSI values on a scale from 0 and 100, where 0 represents the weak or null signal, and 100 represents the strongest signal.

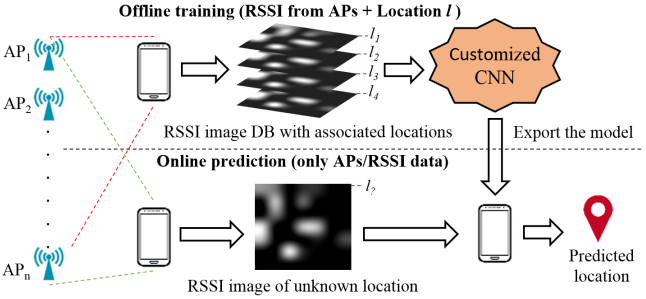

Figure 2: An overview of the *CNN-LOC* framework

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 $[r_1, r_2, ..., r_K]$, denoted by R_N. Therefore, each column vector, $[w_1, w_2, ..., w_N]$ 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 $[l_1, l_2, ..., l_N]$. 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 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.

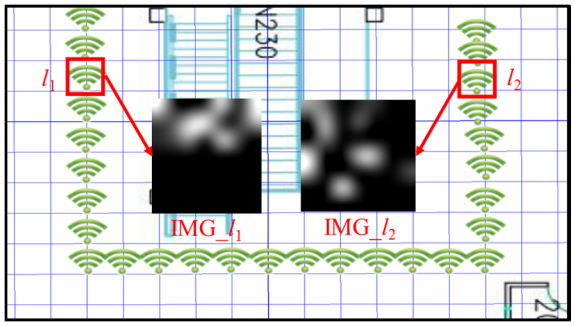

Figure 3: Unique images created for locations l_1 and l_2. The green icons represent locations that are fingerprinted along an indoor path. The two locations shown are 10 meters apart.

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 greyscale image. A collection of these images for all fingerprinted locations forms the RSSI Image Database. To form greyscale 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:

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

The dimension of each *HP* is $1 \times K$. Then, the *HP* matrix is reshaped into a $p \times p$ matrix, which represents a 2D image as shown in figure 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 (*IMG_l1 and IMG_l2*) of size 7×7 created for two unique fingerprints (signatures) associated with two different locations. Each pixel value is scaled on a scale of 0 to 255. The patterns in each of these images will be unique to a location and change slightly as we move along an indoor path.

In equation (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 *CNN-LOC* is discussed next.

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. Identifying the optimal values for the CNN hyperparameters is an empirical process and requires several iterations of experimentation and analysis. The estimated values are also highly dependent on the input dataset. Below, we discuss results of our analysis of CNN hyperparameters for our indoor localization problem domain.

- Number of hidden layers: A large number of hidden layers lead to longer execution times and conversely, fewer hidden layers may produce inaccurate results. We found that 5 layers of [Conv-ReLU] works best for our domain.
- 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 amount 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

Our *CNN-LOC* framework is designed to scale up to larger problem sizes than that handled by most prior efforts. For this purpose, we enhanced *CNN-LOC* 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 decision structure of the framework. Each CNN module has a label that starts with C. The CNN in the first layer (C1) classifies the floor numbers, and then in the next layer, C20 or C21 identify 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. It is important to note that the CNN models in the third layer actually represent two models each, i.e., C30 includes both CNN models for the x and y axis. In this manner, we avoid using the hierarchal classifier twice for each axis.

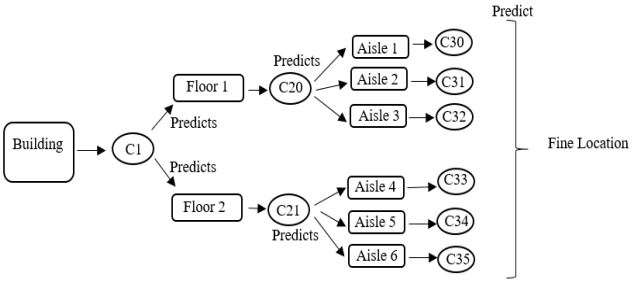

Figure 4: A general architecture for the hierarchical classifier

5. EXPERIMENTS

5.1 Experimental Setup

This section describes the *CNN-LOC* implementation and experimental results. The experiments were conducted on 3 separate indoor paths as described in Table 1. The corridors on the path are divided into a grid and labelled sequentially from 1 to *N*. Each square in the grid has an area of 1 m^2 and represents a "class". This allows us to treat indoor localization as a classification problem for CNN. Figure 5 shows an example of a path covered in the library building 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.

Table 1: Indoor paths used in experiments

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

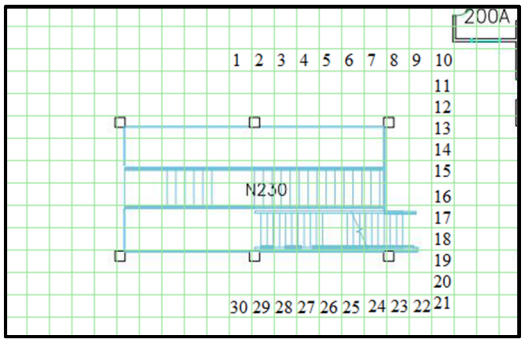

Figure 5: Library building path divided into a grid, with squares along the path labeled sequentially from 1 to 30

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 pre-processed as described in the previous section for the CNN model. The entire data set is split into training and testing samples, so we can check how well our models perform. We used $1/5^{th}$ of the total samples for testing and $4/5^{th}$ of the samples were used for training. Also, we implemented different CNN models for location estimation along different axes. Thus, we use a dedicated CNN model to predict the x coordinate of a location. Similarly, a separate CNN model predicts the y coordinates. The output from these models is combined to get the final results.

5.3 Experimental Results

We compared our *CNN-LOC* 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 points, which are then used for regression. The second work is based on the KNN technique from [8], which is a non-parametric 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 very 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 discusses 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 *CNN-LOC* with that reported by other indoor localization techniques.

5.3.1 Indoor Localization Accuracy Comparison

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 taken to create the training dataset. The distance between each of the green dots is 1 meter. In the training dataset, each green dot is converted into an image. The testing phase consists of the user walking along this path, and the red lines in Figure 6 show 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 *CNN-LOC* perform much better. This is likely because KNN and SVR are both regression based techniques where the prediction is impacted by neighboring data points. In cases where the sampled points are very close to each other, there may not be enough variation across neighboring samples for the regression-based techniques to work properly. 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. From figure 6, it is evident that *CNN-LOC* produces stable predictions for the Clark path.

Figure 7 shows a bar graph that summarizes the average location estimation error for the various techniques on the three different indoor paths considered. We found that the KNN approach is the least reliable among all techniques with a mean error of 5.5 meters 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 *CNN-LOC*. 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 to 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 *CNN-LOC* framework is the most reliable framework with the lowest mean error of less than 2 meters.

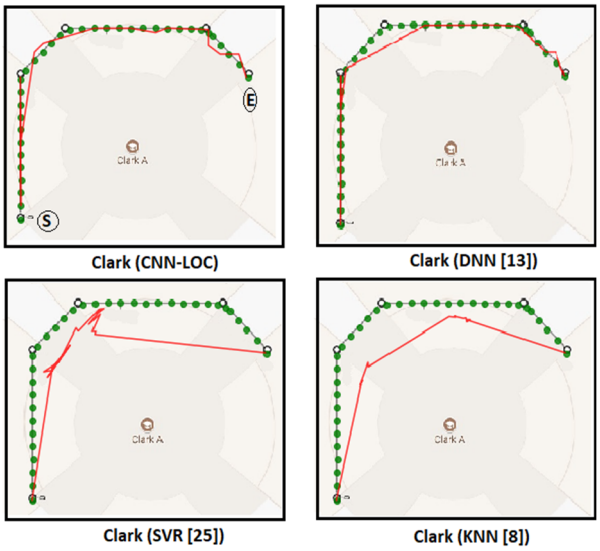

Figure 6: Path traced using different techniques

5.3.2 CNN-LOC Scalability Analysis

We discuss results for the hierarchal *CNN-LOC* (Section 4.5) here. We consider a scenario when *CNN-LOC* 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 meters. 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.

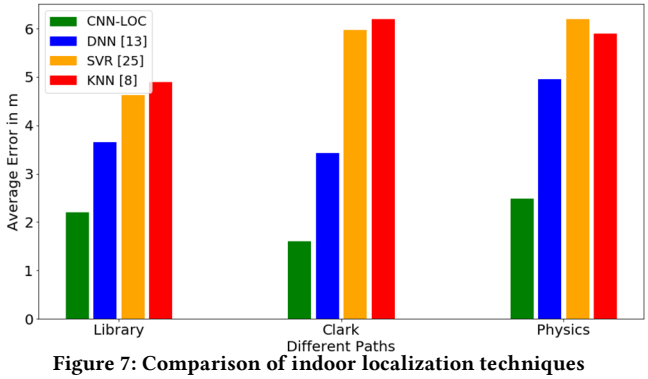

Figure 7: Comparison of indoor localization techniques

We first analyzed the accuracy of predictions, for our *CNN-LOC* framework with and without the hierarchical classifier. For the first and second layer of the hierarchical classifier (shown in Figure 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 and the non-hierarchal 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 non-hierarchical *CNN-LOC* 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 *CNN-LOC* model only takes 2.42ms to make a prediction on average, whereas the non-hierarchical *CNN-LOC* takes longer (3.4ms). Thus, the 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.

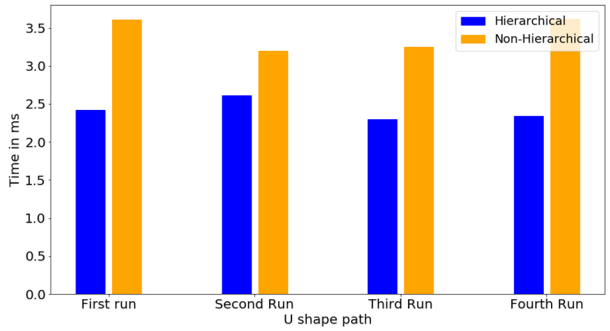

Figure 8: Execution time for Hierarchical CNN

5.3.3 Accuracy Analysis with Other Approaches

Our experimental results in the previous sections have shown that *CNN-LOC* 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 to 7 meters. 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 meters. However, [14] and [15] use special equipment to capture CSI and cannot be used with mobile devices. In contrast, our proposed *CNN-LOC* 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 pre-exists.

6. CONCLUSIONS

In this paper, we presented the *CNN-LOC* framework 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 meters. *CNN-LOC* 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. *CNN-LOC* represents a promising framework that can deliver reliable and accurate indoor localization for smartphone users.

ACKNOWLEDGEMENTS

This research was supported by the National Science Foundation (NSF) under grant number ECCS-1646562, and endowments from the Monfort foundation and Rockwell-Anderson foundation.

REFERENCES

[1] "How Google Maps Makes Money", 2017 [Online] https://www.investopedia.com/articles/investing/061115/how-does-google-maps-makes-money.asp [Accessed: 1 Dec 2017]
[2] "Target and Retailers Using Hybrid Indoor Location Tech to Enable New Customer Experiences" [online] http://www.indoorlbs.com/new-blog-1/2015/11/30/target-and-other-retailers-using-indoor-location-tech [Accessed: 3 Dec 2017]
[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: http://www.ubisense.net/ [Accessed: 1 Dec 2017]
[5] G. Jin, X. Lu, and M. Park, "An indoor localization mechanism using active RFID tag," SNUTC, 2006.
[6] Z. Chen, and C. Wang, "Modeling RFID signal distribution based on neural network combined with continuous ant colony optimization," Neurocomputing, vol. 123, pp. 354-361, 2014.
[7] J. Torres-Sospedra et al., "UJIIndoorLoc: A new multi-building and multi-floor database for WLAN fingerprint-based indoor localization problems," IPIN, 2014.
[8] S. Pasricha, V. Ugave, Q. Han and C. Anderson, "LearnLoc: A framework for smart indoor localization with embedded mobile devices," CODES+ISSS, 2015.
[9] L. Deng, "A tutorial survey of architectures, algorithms, and applications for deep learning," APSIPA, 2013.
[10] G. Borriello et al., "Walrus: wireless acoustic location with room-level resolution using ultrasound," Mobisys, 2005.
[11] C. Yang, and H. R. Shao, "WiFi-based indoor positioning," in IEEE Communications Magazine, vol. 53, no. 3, pp. 150-157, 2015.
[12] P. Bahl, and V. Padmanabhan, "RADAR: An in-building RFbased user location and tracking system," INFOCOM, 2000.
[13] W. Zhang et al., "Deep Neural Networks for wireless localization in indoor and outdoor environments," Neurocomputing, vol. 194, pp. 279-287, 2016.
[14] X. Wang et al., "DeepFi: Deep learning for indoor fingerprinting using channel state information," WCNC, 2015.
[15] H. Chen et al., "ConFi: Convolutional Neural Networks Based Indoor Wi-Fi Localization Using Channel State Information," IEEE Access, pp. 18066 – 18074, vol. 5, 2017.
[16] Y. Hua, et al., "Deep Belief Networks and deep learning," ICIT, 2015.
[17] "Stanford CNN tutorial" [Online] Available: http://cs231n.github.io/convolutional-networks [Accessed: 1 Dec 2017]
[18] D. P. Kingma et al., "Adam: a Method for Stochastic Optimization", ICLR, 2015.
[19] "RMSProp", [Online] http://www.cs.toronto.edu/~tijmen/csc321/slides/lecture_slides_lec6.pdf/ [Accessed: 1 Dec 2017]
[20] J. Duchi et al., "Adaptive Subgradient Methods for Online Learning and Stochastic Optimization", JMLR, vol. 12, pp. 2121–2159, 2011.
[21] Y. LeCun et al., "Gradient-based learning applied to document recognition," Proc. IEEE, vol. 86, no. 11, pp. 2278–2324, Nov. 1998.
[22] A. Krizhevsky et al., "ImageNet classification with deep convolutional neural networks", NIPS, 2012.
[23] J. Benesty, J. Chen, Y. Huang, I. Cohen, "Pearson Correlation Coefficient", in Topics in Signal Processing, Springer, vol. 2, 2009.
[24] G. Styan, "Hadamard products and multivariate statistical analysis", In Linear Algebra and its Apps., Elsevier, vol. 6, pp. 217-240, 1973.
[25] Y. K.Cheng, H. J. Chou and R. Y. Chang, "Machine-Learning Indoor Localization with Access Point Selection and Signal Strength Reconstruction", VTC, 2016.
[26] "IndoorAtlas" [Online] http://www.indooratlas.com/ [Accessed: 1 Dec 2017]
[27] V. Rausch, et al., "Learning a deep neural net policy for end-to-end control of autonomous vehicles," ACC, 2017.
[28] C. Langlois, S. Tiku, S. Pasricha, "Indoor localization with smartphones", CE, 2017.
[29] S. Tiku, S. Pasricha, "Energy-Efficient and Robust Middleware Prototyping for Smart Mobile Computing", RSP, 2017.
[30] A. Khune, S. Pasricha, "Mobile Network-Aware Middleware Framework for Energy-Efficient Cloud Offloading of Smartphone Applications" CE, 2017.
[31] B. Donohoo, C. Ohlsen, S. Pasricha, "A Middleware Framework for Application-aware and User-specific Energy Optimization in Smart Mobile Devices", Journal of Pervasive and Mob. Comp., vol. 20, pp. 47-63, 2015.
[32] B. Donohoo, C. Ohlsen, S. Pasricha, C. Anderson, Y. Xiang, "Context-Aware Energy Enhancements for Smart Mobile Devices", IEEE Trans. on Mob. Comp. (TMC), Vol 13, No. 8, pp. 1720-1732, 2014.