ResLoc: Deep Residual Sharing Learning for Indoor Localization with CSI Tensors

Xuyu Wang, Xiangyu Wang, and Shiwen Mao
Department of Electrical and Computer Engineering, Auburn University, Auburn, AL 36849-5201
Email: {xzw0029, xzw0042}@auburn.edu, smao@ieee.org

***Abstract*—Wi-Fi based indoor localization has attracted great interest due to its ubiquitous access in many indoor environments. In this paper, we propose ResLoc, a deep residual sharing learning based system for indoor localization with channel state information (CSI) tensor data. We first introduce CSI data in wireless systems and show how to build CSI tensors for indoor localization. Then, we present the design of ResLoc, which employs dual-channel, bi-modal CSI tensor data to train the deep network using the proposed deep residual sharing learning in the offline phase. In the online test phase, we use newly received CSI tensor data to estimate the location of the mobile device based on an enhanced probabilistic method. The experimental results show that the proposed ResLoc system can obtain submeter level accuracy with a single access point.**

***Index Terms*—Fingerprinting; deep learning; deep residual sharing learning; 5GHz Wi-Fi; Channel state information.**

I. Introduction

With the remarkable developments in mobile devices and wireless technologies, location-based services (LBS) for Internet of Things (IoT) has become a research hotspot [1]–[3]. High precision location information is indispensable to fulfill the requirements of such applications. Considering the fact that wireless propagation is highly complex in the indoor environment, indoor localization with radio frequency (RF) signals faces great challenges, which draw considerable efforts from researchers.

Among the various technologies, fingerprinting-based indoor localization is a particularly important one, which establishes a database with a large amount of measurements of RF signals in the offline phase, and then computes the position of a mobile device by comparing the newly received RF data with that stored in the database. Due to the low hardware requirements and wide availability (e.g., from laptops, smartphones, or smartwatches), the received signal strength (RSS) is used as fingerprint in many Wi-Fi based fingerprinting systems. Radar is the first RSS-based fingerprinting scheme that adopts a deterministic location estimation method [4]. Horus is another RSS-based fingerprinting scheme, which utilizes a probabilistic location estimation method based on K-nearest-neighbor (KNN) [5]. Moreover, various machine learning methods can be utilized for good localization accuracy, such as neural networks, support vector machines (SVM), and compressive sensing [6]. However, the RSS-based techniques are constrained by two inherent disadvantages [1]: (i) even for the same location, the RSS value usually exhibits high variability due to the complex indoor propagation environment; (ii) RSS is a coarse representation of the wireless channel, where useful subcarrier level information is missing.

978-1-5386-3531-5/17/$31.00

Compared with RSS, channel state information (CSI) represents fine-grained channel information, including subcarrier-level channel measurements in orthogonal frequency division multiplexing (OFDM) systems. In addition, CSI is found to be much more stable than RSS for a given location [7]. Such CSI information can be extracted with open-source device drivers for some chipsets, such as the Intel Wi-Fi Link 5300 network interface card (NIC) [8] and the Atheros AR9580 chipset [9]. Exploiting CSI information, several fingerprinting systems are proposed to achieve better localization performance. For example, FIFS exploits a weighted average of CSI amplitudes over three antennas to achieve fine-grained localization [10], while CSI amplitudes and calibrated phase information are exploited in DeepFi [11] and PhaseFi [12], respectively. These two schemes collect CSI from the subcarriers at all the three antennas and generate fingerprints with a deep autoencoder network. Moreover, to improve the localization accuracy, BiLoc is proposed to exploit bimodal CSI data, including average CSI amplitude and phase difference, with a bimodal deep autoencoder network [13]. Although these deep learning based localization systems can achieve good localization performance, they need to build a database to store extracted features as fingerprints for *every* training location, which translates to more training time and storage requirements.

In this paper, we use bimodal CSI tensor data [14], including estimated angles of arrival (AOA) and CSI amplitude information, that are obtained from the 5GHz WiFi NIC. We find AOA and CSI amplitude are quite stable, which are effective features for fingerprinting based indoor localization. Moreover, AOA and amplitude information are complementary to each other under different indoor environments. For example, when the line-of-sight (LOS) component is strong, the amplitude information can be useful for high localization accuracy. On the other hand, if the LOS signal is blocked (e.g., by a wall) and so the amplitude will be greatly weakened, the estimated AOA values will help to strengthen the localization accuracy. Moreover, we present a new *deep residual sharing learning* technique for improving the training capacity with dual-channel, bimodal CSI tensor data. The residual learning method has been successfully applied for image recognition [15]–[17]. The proposed deep residual sharing learning method is different from the original deep residual

learning framework, in that it shares the residual function. Moreover, we can stack many residual blocks to increase the depth of the deep network, thus achieving higher learning and representation ability. The proposed method only requires for training one group of weights for *all* training locations, as a classification problem in statistical learning, thus significantly reducing the storage requirement.

In particular, we present ResLoc, a deep **Res**idual sharing learning for indoor **Loc**alization with CSI Tensors. In ResLoc, we first construct a CSI tensor including three images, each of which is 30 $\times$ 30 with estimated AOA values and CSI amplitude values. Specifically, we construct two images from estimated AOA values between antennas 1 and 2, and antennas 2 and 3, respectively. The third image is constructed with CSI amplitude values from one antenna. Thus, we can obtain 33 CSI tensors from 990 consecutively received packets from a training location. Moreover, we consider dual-channel CSI tensor data, where the difference between the two CSI tensors is that they use different amplitude information from different antennas when creating the third image. In ResLoc, we use CSI amplitude from antenna 1 and antenna 2, respectively, to construct the dual-channel CSI tensors. For offline training, all the constructed dual-channel CSI tensors from all training locations are used to train the weights of the deep network with the proposed deep residual sharing learning architecture, which includes the input block, the residual block, and the output block. We propose a new design for the residual block, where the residual functions for the two channels are shared, which is effective in exploiting the CSI tensor data. Moreover, we analyze the proposed deep residual sharing learning framework for forward and backward propagation. In the online stage, we use newly received CSI tensor data to compute the location of the mobile device based on an enhanced probabilistic method.

In the remainder of the paper, we introduce the preliminaries and CSI tensor construction in Section II. We present the ResLoc design in Section III and our performance evaluation in Section IV. Section V summarizes this paper.

II. Preliminaries and CSI Tensor Construction

For a Wi-Fi OFDM channel, we denote Ξ_i as the CSI of subcarrier i, which is a complex value given by

$$\Xi_i = |\Xi_i| \exp(j\angle\Xi_i), \tag{1}$$

where $|\Xi_i|$ and $\angle\Xi_i$ are the amplitude response and phase response from subcarrier i, respectively. Besides amplitude information, ResLoc system also uses phase difference information between two adjacent antennas to build the CSI tensor, which is shown to be quite stable in prior work [13], [14], [18]. ResLoc uses the Intel 5300 NIC, from which 30 out of the 56 subcarriers of a 20 MHz or 40 MHz channel can be extracted for a received packet.

To construct a CSI tensor, we first compute bimodal CSI data including estimated AOAs and CSI amplitudes that are obtained from the 5GHz IEEE 802.11n NIC. Besides the amplitude information from the three antennas, it is also easy to estimate the corresponding AOA values for each subcarrier and each received packet as in our prior work, the BiLoc system [13]. Then, for every 30 packets and the corresponding CSI data from the 30 subcarriers, we can construct an order-3 CSI tensor. Specifically, the CSI tensor consists of three images, each of which is 30 $\times$ 30. Each row corresponds to 30 packets and each column corresponds to the 30 subcarriers. The elements are the estimated AOA values and CSI amplitude values. The first two images contain the estimated AOA values, derived from the CSI phase difference data between antennas 1 and 2, and antennas 2 and 3, respectively. The third image consists of CSI amplitude information from one of the antennas. Thus, for 990 packets received from a training or test location, we can construct 33 CSI tensors. Furthermore, we use dual-channel CSI tensor data in ResLoc. For the two CSI tensors in the two channels, they have the same AOA images, but the CSI amplitude images are from different antennas. In the ResLoc system, we use the CSI amplitude information from antenna 1 and antenna 2, respectively, to construct the dual-channel CSI tensors.

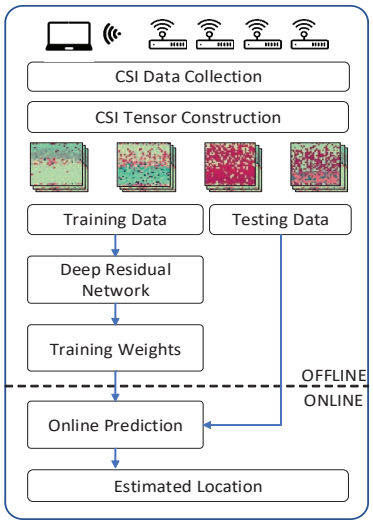

Fig. 1. The ResLoc system architecture.

The CSI tensors will be used as input to ResLoc for training and location estimation. There are three advantages of using CSI tensors. First, using CSI tensors can strengthen the performance of the deep network for solving the classification problem for indoor localization. Moreover, all subcarrier information from all packet samples can be fully exploited with the CSI tensor, which contains rich frequency and time features of the propagation channel. We can thus extract more useful features from the CSI tensor for higher accuracy. Third, we leverage bimodal CSI data including the estimated AOAs and CSI amplitude values for indoor localization, which are complementary to each other under different indoor environments.

III. The ResLoc System

A. ResLoc System Architecture

The ResLoc design is shown in Fig. 1, which consists of one transmitter (i.e., a mobile device) and one receiver, which

is an access point. Both devices are equipped with the Intel 5300 NIC. To collect CSI data, the transmitter is set to the *injection* mode, and the receiver operates in the *monitor* mode. The Intel 5300 NIC reports CSI from 30 subcarriers from each of the three antennas. With the collected CSI data, we construct the dual-channel CSI tensors with estimated AOAs and CSI amplitude values. ResLoc is a fingerprinting based method, which includes an offline training phase and an online location prediction phase. In the training phase, we propose a deep residual sharing learning framework to obtain the optimal weights of the deep residual network, using dual-channel, bimodal CSI tensors. For online location prediction, we utilize newly received CSI tensor data to compute the location of the mobile device with an enhanced probabilistic approach.

ResLoc is very different from traditional fingerprinting based methods, which builds a database for *every* training location with raw data or training features as fingerprints. In fact, ResLoc only requires for training one set of weights for the deep residual network for *all* training locations, which is similar to regression or classification problem in statistical learning. Apparently, this method can significantly reduce the amount of stored data. Furthermore, the proposed deep residual sharing learning also contributes to improving the robustness of the system, which can effectively extract and represent the rich features of CSI data.

B. Offline Training

We propose a deep residual sharing learning framework for training the deep network with dual-channel, bi-modal CSI tensors, which includes an input block, a residual block, and an output block, as shown in Fig. 2.

1) Input Block: In the input block, the bi-modal CSI tensor data is processed by four different layers, namely the Convolution2D layer, the batch normalization layer, the activation layer, and the max pooling layer. The goal is to obtain the local dependency and scale invariant features from bi-modal CSI tensor. Furthermore, the input block can exploit more abstract representations of the input CSI tensor data from lower layers to higher layers, which can improve feature extraction of CSI tensor data. We discuss these four layers in the input block in the following.

The *Convolution2D layer* is to obtain feature maps within local regions of the input CSI tensor or the previous layer's feature maps with several convolution kernels. In fact, each data element of a feature map is connected with the local data in the previous layer. Moreover, by using different convolution kernels we can produce many feature maps. Let Θ_i^l denote feature map i in layer l, defined as

$$\Theta_i^l = \sum_{m \in S_{l-1}} \omega_{im}^l * \Theta_m^{l-1} + b_i^l, \tag{2}$$

where ω_{im}^l is the convolutional kernel to generate feature map i in layer l, b_i^l is the bias of feature map i in layer l, and S_{l-1} is the set of feature maps in layer $(l-1)$ connected to the current feature map, which is the same for different m due to local weights sharing. Convolution with weights sharing can improve the efficiency of training the deep network.

The *batch normalization layer* adjusts the input distribution for different layers and thus alleviates the problem of Internal Covariance Shift (i.e., as the data flow propagates from different layers in the deep network, the distribution of input will be shifted, thus reducing the learning capacity) [19]. In the batch normalization layer, the input data is normalized such that it follows a zero mean and unit standard deviation distribution, where the estimated mean and variance are obtained by each mini-batch. Then, to improve the representation ability in the deep network, the normalized data is shifted and scaled. The batch normalization for the k_{th} input data x_k is as follows.

$$y_k = \gamma \cdot \frac{x_k - u_B}{\sqrt{\sigma_B^2 + \epsilon}} + \beta, \tag{3}$$

where u_B and σ_B^2 are the the mean and variance of the mini-batch, respectively, ϵ is the small constant value to avoid numerical problems, γ and β are the scale and shift parameters, respectively, which are learned by training. Batch normalization is useful for avoiding overfitting in training, which can achieve the same goal as Dropout.

The *activation layer* is employed to avoid obtaining trivial linear combinations of input data, which can detect nonlinear features. Traditional nonlinear activation functions mainly use either the `sigmoid`, i.e., $\sigma(x) = \frac{1}{1+\exp(-x)}$, or the `tanh`, i.e., $\tanh(x) = 2\sigma(x) - 1$, functions. In ResLoc, we adopt the rectified linear unit (i.e., `ReLU`) as the activation function with the expression $ReLu(x) = \max(x, 0)$, which can keep the positive part and suppress the total negative part to zero [20]. For training in deep convolution neural networks, the `ReLU` function can achieve faster training than traditional `sigmoid` and `tanh` functions. Moreover, it can also exploit sparse representations in the hidden units and can facilitate effective training without pre-training.

The *max pooling layer* can reduce the resolution of feature maps by downsampling over a local neighborhood in the feature maps of the previous layer. It is invariant to distortions and small shifts in the inputs. Moreover, it also improves the robustness of the deep network. The feature maps in the previous layer are pooled over a local temporal neighborhood with the max pooling function, as

$$\Theta_{ij}^{l+1} = \max_{k \in \mathcal{S}_j^l} \left\{ \Theta_{ik}^l \right\}, \tag{4}$$

where $\mathcal{S}_j^l$ is the set of pooling region for the jth value in feature map i in layer l and Θ_{ik}^l is the kth value of feature map i in layer l. Other methods such as mean or sum pooling functions can also be used here for reducing training time.

2) Residual Block: In the residual block, we propose a novel *deep residual sharing learning* approach for improving the training capacity with dual-channel CSI tensor data. The proposed method is different from the original deep residual learning unit, with an additional sharing of the residual functions. Moreover, we can stack as many residual blocks as needed, for increasing the depth of the deep network, in

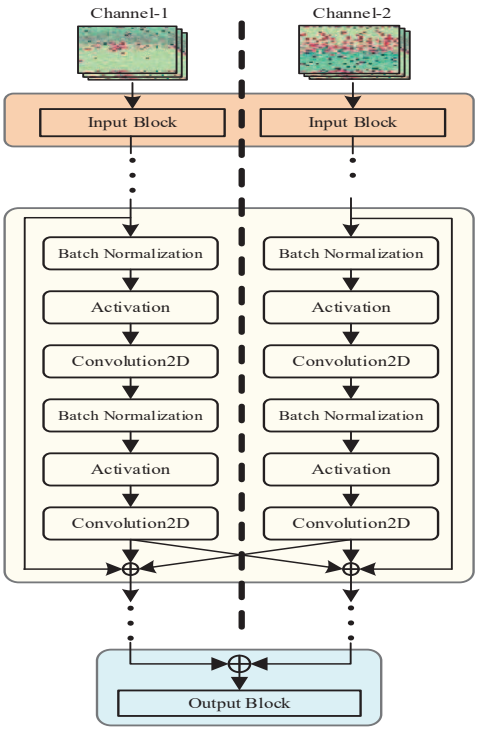

Fig. 2. Deep residual sharing learning for offline training.

order to achieve a higher learning and representation ability. The main idea of residual learning [16], [17] is that, instead of learning the underlying mapping $H(x)$ by using a few stacked layers, we can learn the residual function $F(x) = H(x) - x$. Thus, the original mapping becomes $F(x)+x$, where x is implemented by identity mapping with the shortcut connection. Thus, it is easier for training very deep networks using residual learning. We implement the proposed deep residual sharing learning by sharing the residual functions between the two channels of input data, as shown in Fig. 2. Furthermore, the residual function includes two parallel convolution operations, each of which includes the batch normalization layer, the activation layer, and the convolution2D layers. These are implemented in the same fashion as in the input block.

We next provide an analysis of deep residual sharing learning for forward and backward propagation. Denote x_k^1 and x_k^2 as the input data from channel 1 and channel 2 for the k_{th} residual block, respectively. Let R be the residual function with two 3×3 convolution layers. From Fig. 2, we have

$$\begin{cases} x_{k+1}^1 = x_k^1 + R(x_k^1) + R(x_k^2) \\ x_{k+1}^2 = x_k^2 + R(x_k^2) + R(x_k^1), \end{cases} \tag{5}$$

for the $(k+1)_{th}$ residual block in channel 1 and channel 2, respectively. We then recursively obtain x_K^1 and x_K^2 for the K_{th} residual block in channel 1 and channel 2, respectively, given as

$$\begin{cases} x_K^1 = x_k^1 + \sum_{i=k}^{K-1} R(x_i^1) + \sum_{i=k}^{K-1} R(x_i^2) \\ x_K^2 = x_k^2 + \sum_{i=k}^{K-1} R(x_i^2) + \sum_{i=k}^{K-1} R(x_i^1). \end{cases} \tag{6}$$

With the above equations on forward propagation, we find that the output x_K^1 and x_K^2 shares the same residual function, which can be represented by the summation of preceding residual functions and adding input x_i^1 or x_i^2, respectively. This reduces the error of gradient propagation. Moreover, it is easier to train the shared residual functions, which are forced to zero when the identity mappings are optimal.

Next, let L be the loss function for backward propagation. Following the chain rule of backpropagation, we have

$$\begin{cases} \frac{\partial L}{\partial x_k^1} = \frac{\partial L}{\partial x_K^1}\left(1 + \frac{\partial}{\partial x_k^1}\left(\sum_{i=k}^{K-1}(R(x_i^1) + R(x_i^2))\right)\right) \\ \frac{\partial L}{\partial x_k^2} = \frac{\partial L}{\partial x_K^2}\left(1 + \frac{\partial}{\partial x_k^2}\left(\sum_{i=k}^{K-1}(R(x_i^2) + R(x_i^1))\right)\right). \end{cases} \tag{7}$$

With the the above equations on backward propagation, we find that the gradients $\frac{\partial L}{\partial x_K^1}$ and $\frac{\partial L}{\partial x_K^2}$ are directly propagated back to any shallower input x_k^1 and x_k^2, respectively. Moreover, because the gradients for the shared residual functions $\frac{\partial}{\partial x_k^1}(\sum_{i=k}^{K-1}(R(x_i^1)+R(x_i^2)))$ and $\frac{\partial}{\partial x_k^2}(\sum_{i=k}^{K-1}(R(x_i^2)+R(x_i^1)))$ are not always -1, the gradients $\frac{\partial L}{\partial x_K^1}$ and $\frac{\partial L}{\partial x_K^2}$ cannot be canceled for the mini-batch with stochastic gradient descent (SGD), thus avoiding the problem of the vanishing gradient. Thus, the proposed deep residual sharing learning can increase the learning capacity and fully exploit the dual-channel CSI tensor data.

3) Output Block: In the output block, we first merge two channel data into a single channel. Then, we implement several basic data operations for the merged data, including batch normalization, activation with ReLU, and max pooling. The main operation in the output block is the fully-connected layer, which employs a standard neural network with one hidden layer to train the output data based on the *softmax* classifier. The input data to the softmax function is an N dimensional vector $z = [z_1, z_2, ..., z_N]$, where N is the number of clusters. Then, the softmax function can map the N dimensional vector to the normalized data $p = [p_1, p_2, ..., p_N]$, that is

$$p_i = \frac{e^{z_i}}{\sum_{r=1}^{N} e^{z_r}}, \quad \text{for } i = 1, 2, ..., N. \tag{8}$$

In addition, we define the loss function as the cross-entropy to measure the difference between the output normalized data and the true label data, that is

$$L = -\sum_{r=1}^{N} y(r) \log(p_r), \tag{9}$$

where $y(r)$ is the true label data for the rth location. Then, we train the parameters in the deep network with the SGD method by minimizing the loss function.

C. Online Test Phase

In the online test phase, a probabilistic method is utilized to estimate the position of the mobile device. Let T denote the number of CSI tensors from one position, and p_{ij} be the output result of the deep network with CSI tensor j for location i. The matrix $\boldsymbol{P}$ is the prediction output of the deep network

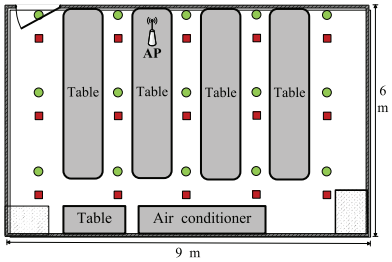

Fig. 3. Layout of the laboratory, where training locations are marked by red squares and testing locations are marked by green dots.

with T CSI tensors for N training locations, given by

$$\boldsymbol{P} = \begin{bmatrix} p_{11} & p_{12} & p_{13} & \cdots & p_{1T} \\ p_{21} & p_{22} & p_{23} & \cdots & p_{2T} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ p_{N1} & p_{N2} & p_{N3} & \cdots & p_{NT} \end{bmatrix}. \quad (10)$$

To reduce the variance of the output results, T output data for every location are averaged out. Thus, we can obtain a vector $\bar{P} = [\bar{p}_1, \bar{p}_2, ..., \bar{p}_N]$, where $\bar{p}_i$ is the mean of the output vector $[p_{i1}, p_{i2}, ..., p_{iT}]$ in row i.

Finally, we can compute the location of the mobile device as a weighted average of all the N training locations, that is

$$\hat{L} = \sum_{i=1}^{N} l_i \times \bar{p}_i, \quad (11)$$

where l_i is the ith training location.

IV. Experimental Study

A. Experiment Configuration

To evaluate the performance of ResLoc, we implement it with off-the-shelf 5GHz WiFi devices. In order to collect CSI data, a desktop computer and a Dell laptop are used as the access point and mobile device, respectively. Both computers are equipped with an Intel 5300 NIC, running Ubuntu desktop 14.04 LTS system. To transmit to the desktop, the Dell laptop uses one antenna in the injection mode. The desktop works in the monitor mode to receive data using all the three antennas. The distance between two adjacent antennas at the Desktop is set to 2.68cm, which is a half wave length for 5.58 GHz WiFi. Moreover, the PHY is IEEE 802.11n OFDM with QPSK modulation and 1/2 coding rate. To accelerate the training process, we employ in the offline stage of ResLoc Keras with the tensorflow backend on a PC with Intel(R) Core(TM) i7-6700K CPU, and a Nvidia GTX1070 GPU [21].

ResLoc is compared with two existing deep learning based fingerprinting approaches, i.e., BiLoc [13] and DeepFi [11]. We also consider the localization performance of ResLoc using a single channel. For the sake of fairness, the same CSI training dataset and testing dataset are used by all the four schemes. We conduct experiments in two typical indoor environments, including a computer laboratory and a corridor.

A Laboratory Scenario: We set up the first testbed in a 6×9 m^2 computer laboratory in Broun Hall in the Auburn University campus. This laboratory is a cluttered environment. The furniture, computers, and appliances block most of the LOS paths. 15 training locations are shown as red squares in Fig. 3, while the other 15 green dots are testing locations. The distance between two adjacent training locations is 1.8 m. The access point is fixed on the table. We collect CSI data from 1,000 received packets for every training location and test location. Moreover, we set the number of layers of the proposed deep network to 34, which can achieve high localization accuracy with a short training time.

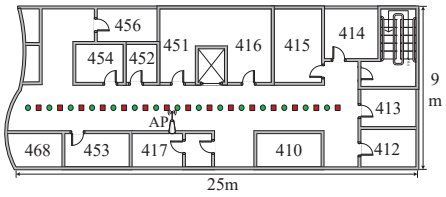

Fig. 4. Layout of the corridor, where training locations are marked by red squares and testing locations are marked by green dots.

A Corridor Scenario: We set up the second testbed in a long corridor in Broun hall, which is 9×25 m^2. The corridor is empty with no obstacles, and thus the LOS path is dominant. We choose 15 training locations and 15 testing locations in a straight line. The distance between two adjacent training locations is 1 m. The red squares are training locations and the rest green dots are testing locations. We set the receiver in the middle of the corridor. 1000 packets are received from every training location and testing location to collect CSI data. The number of layers in the deep network is the same as that in the laboratory scenario.

B. Experimental Results and Discussions

Figure 5 depicts the training loss over epochs of ResLoc for the laboratory and corridor scenarios. To prevent overfitting for the training CSI tensor dataset and reduce training time, the epoch is set to 50. As shown in Fig. 5, the training loss of the corridor curve reaches 0.3 and the training loss of the lab curve stops at about 0.5, when the training is over. With the Nvidia GTX1070 GPU, we observe short training time for both the laboratory and corridor scenarios, which are 608.14 s and 619.35 s, respectively. The test time for the laboratory and corridor scenarios are 0.587 s and 0.647 s, respectively, which are sufficiently fast for realtime location estimation.

Figure 6 presents the cumulative distribution function (CDF) of distance errors across the 15 test positions in the laboratory scenario. Unlike the corridor scenario where the LOS is dominant, the furniture and appliances block most of the LOS paths in this environment. As we can see, the maximum distance errors for ResLoc with two channels and a single channel are about 2.5 m, which is less than that of DeepFi and BiLoc. In addition, the median of distance errors for ResLoc with two channels and a single channel are about 0.89 m, which also outperforms BiLoc and DeepFi by 0.51 m and 0.89 m, respectively. For ResLoc with two channels, the distance error of over 30% testing data is less than 0.3 m. However, there is

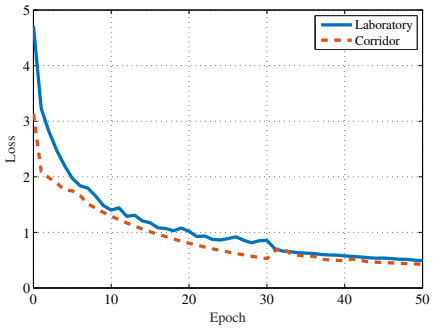

Fig. 5. Training errors for the laboratory and corridor experiments.

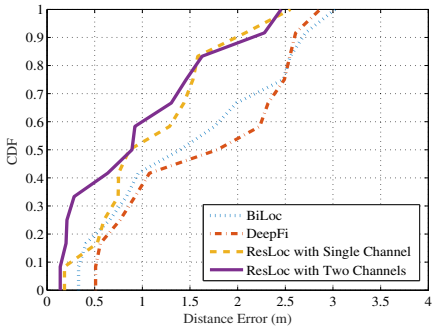

Fig. 6. CDF of localization errors for the laboratory experiment.

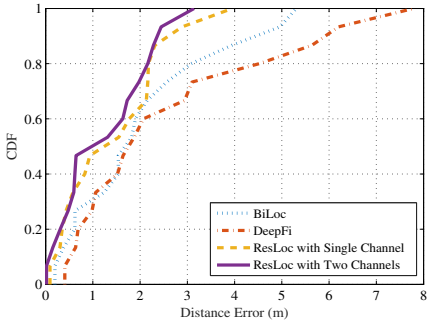

Fig. 7. CDF of localization errors for the corridor experiment.

no data falling within this error range for DeepFi and BiLoc. In summary, based on the proposed deep residual sharing learning, ResLoc with dual-channel tensor data achieves the best performance in this rich multipath scenario.

Figure 7 plots the CDF of localization errors in the corridor scenario. As shown in Fig. 7, the maximum distance error for ResLoc with two channels and a single channel are 3.14 m and 3.95 m, respectively, which are significantly less than that of the other two schemes. In addition, the median of distance errors for ResLoc with two channels, ResLoc with a single channel, BiLoc, and DeepFi are about 0.98 m, 1.24 m, 1.68 m, and 1.75 m, respectively. Thus, ResLoc with two channels also achieves the best performance in this scenario. In addition to higher accuracy, the proposed ResLoc system only requires one set of weights for all the training locations, rather than storing fingerprints for every training location like BiLoc and DeepFi. Furthermore, ResLoc does not need a predefined ratio for fusing the bi-modal data, making it easier to deploy.

V. Conclusions

In this paper, we presented ResLoc, a deep residual sharing learning based system for indoor localization with dual-channel, bimodal CSI tensor data. We presented the design of the ResLoc system, which leveraged dual-channel CSI tensor data to train a deep network using the proposed deep residual sharing learning approach, as well as an enhanced probabilistic method for online location estimation. Although the deep network consisted of 34 layers, it only took 600+ seconds to train and was sufficiently fast for realtime localization. The superior performance of ResLoc was validated with extensive experiments and comparison with several existing deep learning based approaches.

Acknowledgment

This work is supported in part by the NSF under Grant CNS-1702957, and by the Wireless Engineering Research and Education Center (WEREC) at Auburn University.

References

[1] Z. Yang, Z. Zhou, and Y. Liu, "From RSSI to CSI: Indoor localization via channel response," *ACM Computing Surveys*, vol. 46, no. 2, pp. 25:1–25:32, Nov. 2013.

[2] J. Xiong and K. Jamieson, "Arraytrack: A fine-grained indoor location system," in *Proc. ACM NSDI'13*, Lombard. IL, Apr. 2013, pp. 71–84.

[3] Y. Wang, J. Liu, Y. Chen, M. Gruteser, J. Yang, and H. Liu, "E-eyes: Device-free location-oriented activity identification using fine-grained WiFi signatures," in *Proc. ACM Mobicom'14*, Maui, HI, Sept. 2014, pp. 617–628.

[4] P. Bahl and V. N. Padmanabhan, "Radar: An in-building RF-based user location and tracking system," in *Proc. IEEE INFOCOM'00*, Tel Aviv, Israel, Mar. 2000, pp. 775–784.

[5] M. Youssef and A. Agrawala, "The Horus WLAN location determination system," in *Proc. ACM MobiSys'05*, Seattle, WA, June 2005, pp. 205–218.

[6] H. Liu, H. Darabi, P. Banerjee, and L. Jing, "Survey of wireless indoor positioning techniques and systems," *IEEE Trans. Syst., Man, Cybern. C*, vol. 37, no. 6, pp. 1067–1080, Nov. 2007.

[7] S. Sen, B. Radunovic, R. R. Choudhury, and T. Minka, "You are facing the Mona Lisa: Spot localization using PHY layer information," in *Proc. ACM MobiSys'12*, Low Wood Bay, UK, Jun. 2012, pp. 183–196.

[8] D. Halperin, W. J. Hu, A. Sheth, and D. Wetherall, "Predictable 802.11 packet delivery from wireless channel measurements," in *Proc. ACM SIGCOMM'10*, New Delhi, India, Sept. 2010, pp. 159–170.

[9] Y. Xie, Z. Li, and M. Li, "Precise power delay profiling with commodity WiFi," in *Proc. ACM Mobicom'15*, Paris, France, Sept. 2015, pp. 53–64.

[10] J. Xiao, K. Wu., Y. Yi, and L. Ni, "FIFS: Fine-grained indoor fingerprinting system," in *Proc. IEEE ICCCN'12*, Munich, Germany, Aug. 2012, pp. 1–7.

[11] X. Wang, L. Gao, S. Mao, and S. Pandey, "CSI-based fingerprinting for indoor localization: A deep learning approach," *IEEE Trans. Veh. Technol.*, vol. 66, no. 1, pp. 763–776, Jan. 2017.

[12] X. Wang, L. Gao, and S. Mao, "CSI phase fingerprinting for indoor localization with a deep learning approach," *IEEE Internet of Things J.*, vol. 3, no. 6, pp. 1113–1123, Dec. 2016.

[13] ——, "Biloc: Bi-modal deep learning for indoor localization with commodity 5ghz wifi," *IEEE Access*, vol. 5, pp. 4209–4220, 2017.

[14] X. Wang, C. Yang, and S. Mao, "Tensorbeat: Tensor decomposition for monitoring multi-person breathing beats with commodity WiFi," *ACM Transactions on Intelligent Systems and Technology*, 2017.

[15] A. Krizhevsky, I. Sutskever, and G. Hinton, "ImageNet classification with deep convolutional neural networks," in *Proc. Neural Inform. Proc. Syst. 2012*, Lake Tahoe, NV, Dec. 2012, pp. 1106–1114.

[16] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE CVPR'16*, Las Vegas, NV, June 2016, pp. 770–778.

[17] ——, "Identity mappings in deep residual networks," in *Proc. 2016 European Conf. Computer Vision*. Amsterdam, The Netherlands: Springer, Oct. 2016, pp. 630–645.

[18] J. Gjengset, J. Xiong, G. McPhillips, and K. Jamieson, "Phaser: Enabling phased array signal processing on commodity WiFi access points," in *Proc. ACM Mobicom'14*, Maui, HI, Sept. 2014, pp. 153–164.

[19] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," *arXiv preprint arXiv:1502.03167*, 2015.

[20] V. Nair and G. E. Hinton, "Rectified linear units improve restricted boltzmann machines," in *Proc. 27th international conference on machine learning*, Haifa, Israel, June 2010, pp. 807–814.

[21] M. Abadi, et al., "TensorFlow: A system for large-scale machine learning," in *Proc. USENIX OSDI'16*, Savannah, GA, Nov. 2016.