The following publication J. Guo, I. W. -H. Ho, Y. Hou and Z. Li, "FedPos: A Federated Transfer Learning Framework for CSI-Based Wi-Fi Indoor Positioning," in IEEE Systems Journal, vol. 17, no. 3, pp. 4579-4590, Sept. 2023 is available at https://dx.doi.org/10.1109/JSYST.2022.3230425.

FedPos: A Federated Transfer Learning Framework for CSI-Based Wi-Fi Indoor Positioning

Jingtao Guo, Ivan Wang-Hei Ho, *Senior Member, IEEE*, Yun Hou, *Senior Member, IEEE*, Zijian Li

***Abstract*—This paper proposes FedPos, a federated transfer learning framework together with a novel position estimation method for Wi-Fi indoor positioning. Compared with traditional machine learning with privacy leakage problems and the cloud model trained through federated learning (FL) fails in personalization, the FedPos framework aggregates non-classification layer parameters of models trained from different environments to build a robust and versatile encoder on the cloud server while preserving user privacy. The global cloud encoder can aggregate different classifiers and then construct personalized models for new users through fine-tuning. The proposed framework can be updated incrementally and is highly extensible. Specifically, we exploit channel state information (CSI) as the positioning feature and assess the transferability of a lightweight convolutional neural network (CNN) in unfamiliar environments. We evaluate the performance of our proposed framework and position estimation method in different indoor environments. Our experimental results indicate that the proposed framework can achieve a mean localization error of 42.18 cm in a 64-position living room. They also confirm that FedPos can achieve a 5.22% average localization performance boost and reduce the average model training time by about 34.78% when compared with normal training. By reusing part of the feature extractor layers that are trained from other environments, at least 65% of training data can be saved to achieve a localization performance that is similar to the base model. Overall, the proposed position estimation method can effectively improve localization accuracy as compared with three other existing CSI-based methods.**

***Index Terms*—Wi-Fi Fingerprinting, Indoor Positioning, Channel State Information, Federated Transfer Learning**

I. Introduction

INDOOR localization technology is vital in a broad range of scenarios, such as self-governing tours, emergency services, and smooth handoff in mobile communications [1]. Up till now, many radio frequency-based solutions have been proposed for indoor positioning tasks, such as Wi-Fi, Zigbee, and Bluetooth [2]. Wi-Fi fingerprinting technology is one of the most common methods because of its low cost, large range of adoption, high localization accuracy, and user-friendly operation. For fingerprinting-based Wi-Fi indoor positioning systems, a large amount of them uses received signal strength indicator (RSSI) that can easily be obtained from Wi-Fi receivers, such as smartphones, laptops, and tablets [3], [4]. However, there are two shortcomings in RSSI. First, the feature information used for indoor localization is insufficient. For instance, only one RSSI can be derived from a transmission packet at one time when using a single router, which results in the need for a large number of access points (APs) for accurate positioning. For example, over 50 APs were used in [5] and six APs were adopted in [6] for an indoor area of around 2756 m^2 and 336 m^2, respectively. Since the indoor environment is dynamically changing, it may not be feasible to deploy a large number of APs for indoor localization [7]. Besides, there may only be one router installed in small-area spaces, such as classrooms, offices, or small stores, which do not allow us to perform trilateration based on RSSI. RSSI is also unstable since it is affected by multipath fading, which makes it a fickle and coarse-grained feature [8]. To tackle these issues mentioned above, we exploit channel state information (CSI) as the localization feature. In comparison with RSSI, CSI is a finer-grained channel response derived from the port physical layer (PHY), which can provide both amplitude and phase information in frequency domain [8]. However, only a few specific hardware/software combinations are allowed to extract CSI data.

This work was supported in part by the Key-Area Research and Development Program of Guangdong Province (2020B 090928001); and by The Hong Kong Polytechnic University (Project No. 4-ZZMU).

J. Guo and I. W.-H. Ho are with the Department of Electronic and Information Engineering, The Hong Kong Polytechnic University, Hong Kong (email: jingtao.guo@connect.polyu.hk; ivanwh.ho@polyu.edu.hk)

Z. Li is with the Department of Electronic and Computer Engineering, The Hong Kong University of Science and Technology, Hong Kong (email: zijian.li@connect.ust.hk)

Y. Hou is with the Department of Computing, The Hang Seng University of Hong Kong, Hong Kong (email: aileenhou@hsu.edu.hk)

A. Related Works and Motivations

Early in 2011, the Linux 802.11n CSI Tool was released, which was built for the Intel IWL5300 network interface card (NIC) [9]. After that, most of the CSI-based research was performed based on this platform. It is one of the cheapest and most stable ways to extract CSI data from Wi-Fi routers. For example, [10] introduced an algorithm named time-reversal resonating strength (TRRS) for indoor positioning, which can achieve high position and tracking accuracy. However, [11] found that the localization performance of the original TRRS method may suffer in a dynamic environment. To improve the localization performance on a daily basis, they proposed three quick remedies, namely (1) Continuous Fingerprint Appending (CFA), (2) Fingerprint Averaging (FA), and (3) Weighted Fingerprint Averaging (WFA). While all these approaches either require relatively large database storage or only include coarse information and lose some fine-grained features. After that, [12] proposed a Wi-Fi fingerprinting scheme termed CSI-MIMO for indoor localization. This scheme leveraged both CSI amplitude and phase information as the localization feature and estimated the unknown location based on a maximum posterior probabilistic method. After that, the DeepFi system

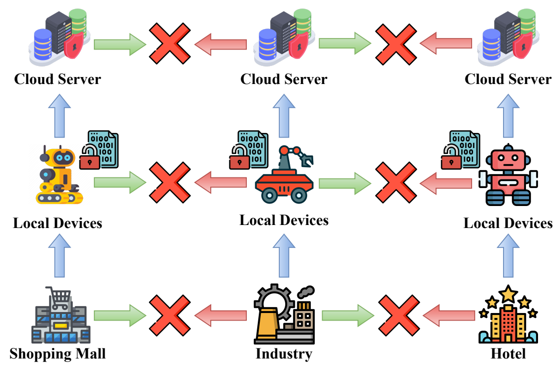


Fig. 1: Data isolation and personalized training issues in indoor localization.

[13] proposed another novel position estimation method which used the weighted average of all fingerprinted reference points (RPs) to estimate the final position. The ConFi scheme [4] further introduced a weighted centroid method to estimate the unknown location. It also proves that CNN is more suitable for indoor localization than a fully connected neural network (NN) in terms of feature extraction capability. However, the position estimation algorithms in these schemes only consider either arbitrary testing points or fingerprinted RPs during the location estimation stage, whereas target devices may locate at either of them in practical localization.

Later in 2019, Nexmon CSI [14] was released, which makes it possible to extract CSI data using Raspberry Pi 3B+/4B. Therefore, it further facilitates CSI-related research. The onboard Wi-Fi chip of Raspberry Pi 4B can work on the 802.11ac protocol. Therefore, more CSI data can be included in one packet from one pair of transceivers when compared with IWL 5300 NIC which works on 802.11n. Moreover, the use of the 5 GHz band can also provide a more stable Wi-Fi signal [15]. Hence, Raspberry Pi 4B has the potential of providing higher indoor localization accuracy. We use CSI data collected from the Raspberry Pi 4B platform in Line of Sight (LOS) and Non-Line of Sight (NLOS) scenarios to evaluate our proposed framework in this work. For modeling the relationship between the position of the target and the dynamically changing signal features, machine learning (ML) and deep learning (DL) techniques are applied.

In general, DL models require a huge amount of data for training so that they can be used for indoor localization in different environments. However, two important issues exist in today's indoor localization applications (see Fig. 1). First, training data for indoor positioning should not be publicly shared between environments, local devices, and cloud servers because of privacy and security considerations [1]. Although there is a vast variety of data in different organizations and institutes for different subjects, sharing this data is not practical owing to security and privacy concerns. In Fig. 1, devices' data stored in different cloud servers cannot be transferred when the local user utilizes distinct products from two different providers. Because of this, it is challenging to train robust models using these data sources. Additionally, China, the US, and the EU have recently implemented various regularizations to enforce the protection of user data [16], [17]. Therefore, it is hard for practical applications to collect an enormous amount of user data for model training. In addition, data misuse problems may also happen when they are shared with third parties. Hence, datasets exist as isolated islands [18], which makes it hard to train a robust model using sufficient data. Federated learning (FL) is introduced to address this issue. This method can facilitate the training of robust localization services without privacy leakage [19]. As an example, [20] leveraged the FL method to build an indoor federated localization scheme based on Wi-Fi fingerprinting. The potential of FL in indoor localization services was also demonstrated in [21]. These works utilize the computational resources of mobile devices to distributedly train a global cloud model for identifying users' indoor positions. After the local model training, these local models will be uploaded to the parameter server to aggregate and build a global model. The global model is updated for all local devices, and daily indoor positioning information for different local users can then be tracked. However, most of the existing works did not consider another critical issue regarding personalization. In Fig. 1, the layouts of the industry and hotel are entirely different, resulting in different moving trajectories for the devices in these environments, and hence the fingerprint layouts. Therefore, users' data are not independently and identically distributed (non-IID) on edge devices. It is hard to directly aggregate local models trained in different environments, and the global model also cannot adequately consider personalized positioning information, which may result in poor localization performance on edge devices [21], [22].

B. Contributions

In this study, we propose a federated transfer learning framework together with a novel position estimation method for fingerprinting-based indoor localization, which is termed as FedPos. This framework aims to tackle the challenge mentioned above and can be easily extended to other application scenarios, e.g., healthcare and image steganalysis. It also allows the crowdsourcing of CSI data. By using the FL method [18] and homomorphic encryption [23] technology, our framework aggregates non-classification layer parameters (parameters for all layers except the final layer) of isolated user models to build a robust cloud encoder without leaking users' private information. When the cloud encoder is built, it can be delivered to a new user to form a personalized model. Different from some existing federated transfer learning frameworks that aggregate all parameters of user models to build a global cloud model as presented in [24], [25]. We only exploit the non-classification layers of user models to form a robust cloud encoder to learn the model representation since the number of parameters for the final layer is different for users with different indoor trajectories. In such cases, combining different classifiers for different indoor positioning tasks becomes convenient, and its transmission cost is reduced. This distributed framework can be updated continuously with new indoor localization data. It is highly extensible and is suitable for personalized model training that requires the protection of privacy and security of user data. Overall, the major contributions of this study are three-fold:

- We propose a novel probabilistic-based method for position estimation. We consider both fingerprinted RPs and arbitrary testing points in the testing stage. Three indoor environments with different sizes are used to validate our proposed method, and the results outperform seven existing position estimation methods.
- We propose FedPos, a federated transfer learning framework for CSI-based indoor positioning with a single AP. It aggregates non-classification layer parameters of trained user models with preserving privacy and builds personalized models for new users through deep transfer learning. This proposed framework exploits FL to build a robust and versatile feature extractor in the cloud server to improve the performance of normal training. As a result, it makes combining different classifiers for performing different local positioning tasks convenient.
- We conduct extensive experiments with the Raspberry Pi platform to evaluate the proposed framework. Our results indicate that FedPos has strong transferability, which can achieve a 5.22% average localization performance boost for a new environment when compared with normal training.

C. Organization

The rest of the paper is organized as follows: In Section II, we introduce the background of CSI, transfer learning, and FL. Then we illustrate our proposed federated transfer learning framework in Section III. After that, we introduce the experimental setup for evaluating the proposed framework and position estimation algorithm in Section IV and present the experimental results in Section V. Finally, Section VI concludes the paper.

II. Background

A. Channel State Information

In recent years, CSI has become an efficient feature in performing indoor positioning [2], [26]. CSI can provide much more information, and it is more stable for constantly received packets at a fixed location [13]. Popular CSI extraction hardware such as the Intel IWL5300 NIC can access 30 subcarriers per packet from one pair of transceivers connected to 2.4 GHz Wi-Fi. Moreover, some new devices such as Raspberry Pi 4B can access up to 256 sub-carriers per packet per transmit-receive antenna pair connected to 5 GHz 802.11ac Wi-Fi, which enables PI 4B to collect CSI data for training a human activity recognition model as presented in [27]. Since there are spatial and temporal correlations between adjacent channels and samples respectively [28], the data from multiple channels can be transformed into a 2-D radio image for extracting color and texture features for DL model training as introduced in [29]. Raspberry Pi 4B can obtain richer CSI data per packet to form CSI images with more multipath information, and CNN can be applied to distinguish these CSI images to perform positioning. This work uses Pi 4B to collect CSI data for training an indoor positioning CNN model. According to [30], CSI is a fine-grained physical-layer measurement that combines the effects of signal scattering and multi-path fading to describe the channel frequency response (CFR) between the access point and the receiver. Many Wi-Fi devices have multiple transmitting and receiving antennas, which produce multiple-input multiple-output (MIMO) wireless networks for high data rate transmission. Based on the orthogonal frequency domain multiplexing (OFDM) method, digital data are encoded on multiple subcarrier frequencies, which allows the transmitted signal to carry more data in parallel. Each subcarrier in one packet and the CSI matrix can be represented in (1) and (2) respectively:

$$C_i = |C_i| e^{j \cdot \theta_i} \tag{1}$$

$$\boldsymbol{C} = \begin{bmatrix} C_{11} & C_{12} & \cdots & C_{1i} \\ \cdots & \cdots & \cdots & \cdots \\ C_{M1} & C_{M2} & \cdots & C_{Mi} \end{bmatrix}, i \in [1, N] \tag{2}$$

where $|C_i|$ and θ_i are the amplitude and phase response of subcarrier i, respectively. N represents the number of subcarriers per packet and M is the total number of packets transmitted between the transmitter and receiver. An environment directly influences the measurements of the amplitude and phase response. Therefore, they can be regarded as positioning features in a specific range. However, the phase responses of CSI are noisier and randomly fluctuating because of the fading effect and frequency offset [4], [13]. Thus, complex signal preprocessing is required before using it as a positioning characteristic. Therefore, only the amplitude response of CSI is utilized in our proposed framework.

B. Federated Transfer Learning

FL [18] is a distributed ML method, which trains cloud models using a large amount of decentralized data collected from mobile devices all over the world. The crucial object is to build a cloud model while preserving user privacy. This ML method can tackle the existing data isolation problem, but it cannot build personalized models. The model is general on the cloud, which means it only learns the coarse features from all clients. Therefore, fine-grained information for a particular user cannot be learned and this may lead to poor performance for a particular participant. This happens because of the data distribution mismatch among different participants. To address this issue, transfer learning is used in our proposed framework to help build a personalized model.

Transfer learning, which adapts the model trained in the source domain to the target domain, was first proposed in 2009 [31] and has been successfully applied and shown outstanding performance in extensive real-world applications. It aims to reduce the data mismatch between different domains. In our proposed framework, network-based deep transfer learning (i.e., fine-tuning) is adopted to build personalized models for new users. As analyzed in [32], CSI data have some common hidden features even though the data packets are obtained from different environments with different sizes and layouts. For instance, the amplitude of adjacent subcarriers maintains a certain level of consistency, which holds within a certain range. These common hidden features are assumed to be

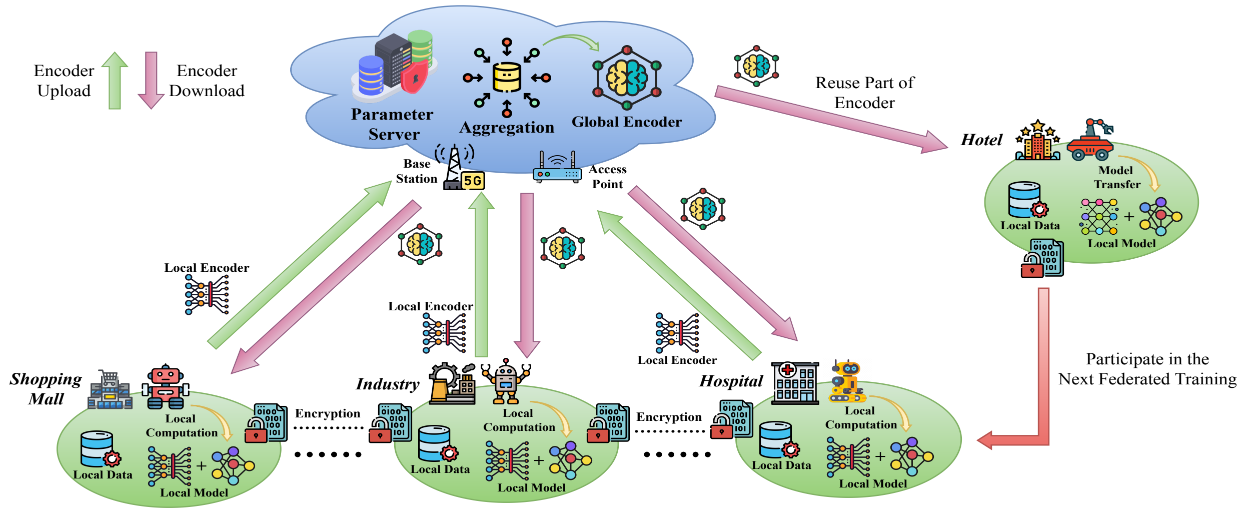


Fig. 2: Overview of FedPos, the proposed federated transfer learning framework.

extracted from the front layers in a CNN model and passed to subsequent layers, which serves as the foundation for deep transfer learning. To improve the training efficiency in a new environment, the front layers of the trained CNN model can be reused to leverage their feature extraction ability. In addition, the indoor multi-path and shadowing effect can be mitigated by tuning the layers after the feature extraction layers. Our proposed framework makes it easy for a new user to access a pre-trained feature extractor as long as there is a network connection in the new environment.

Federated transfer learning considers scenarios with few common samples between different datasets, which usually happen in indoor localization. This method is a crucial extension to the existing transfer learning systems since it solves the data isolation problem in the indoor localization field. As far as we are aware, there is no such framework for indoor localization. The proposed framework can also be adopted in other scenarios that have the same privacy-preserving and personalization requirements.

III. The Proposed Federated Transfer Learning Framework

A. Problem Definition

We denote $\{\mathcal{P}_i\}_{i=1}^N$ and $\{\mathcal{D}_i\}_{i=1}^N$ as N different data owners (that live/work in different environments) and their respective data (CSI data for indoor positioning). We assume that there are no CSI data stored on the cloud server because of the privacy consideration. It is also hard to find a large public CSI fingerprinting dataset for building a robust common model on the cloud server. Conventional deep learning methods use data collected from a new environment to train a personalized model $\mathcal{M}$. We aim to build a cloud feature extractor that takes all participants' data into account with preserving privacy and generates new decoders for new personalized models $\mathcal{M}_{FedTr}$. Our proposed framework can improve the localization performance of the conventional deep learning model by combining it with the cloud feature extractor.

B. Major Idea of the Proposed Framework

Our proposed framework aims to improve the normal training performance for indoor localization under data privacy and security consideration. Fig. 2 demonstrates the overview of our framework. We suppose there are four different environments where four users live/work (i.e., one user lives/works in one environment) and a cloud server, which can be expanded to include more common cases. The following four parts comprise our framework. (1) Each user uses his/her data to train his/her model. (2) All users who participate in this federated transfer learning system then upload non-classification layer parameters of their models to the server to help update a cloud feature extractor by parameters aggregation. (3) The updated cloud feature extractor is delivered to all participants to combine their classifiers for further training. Note that only encrypted model parameters are shared during the uploading and downloading phases. (4) Personalized training can be performed by new users to fine-tune the model by combining the updated cloud feature extractor and the user's classifiers. As there is a significant mismatch between users' data distribution in different environments, transfer learning is adopted to help train a personalized model. This learning method makes the retraining procedure only require a relatively small amount of data (see the right side of Fig. 2). All steps that include sharing model parameters must protect user privacy and security through homomorphic encryption.

The main computational procedure in our proposed framework comes from FL which includes federated model training and parameter sharing in the whole framework. The trained cloud feature extractor is delivered to new users for personalized training. As we can see, the 'encryption lock' in Fig. 2 is used to forbid the direct sharing of user information. This encryption lock can be achieved through the additively homomorphic encryption method. Since this part is out of the scope of this work, we will use real numbers to describe the process of additively homomorphic encryption. It can be extrapolated to weight matrices and bias vectors. Let $\langle \boldsymbol{x} \rangle$ denote any real number $\boldsymbol{x}$ after performing additively homo-

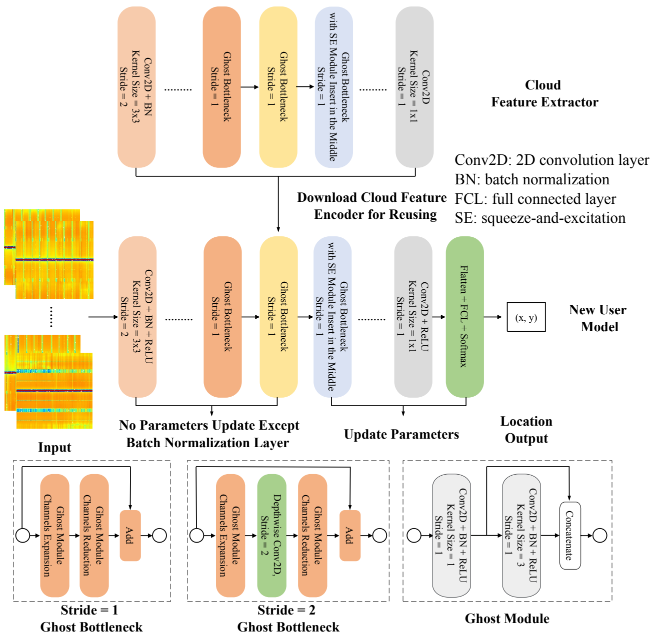


Fig. 3: Model transfer procedure of our proposed framework. Dashed boxes describe the composition of the Ghost Bottleneck and the detail of the Ghost Module.

morphic encryption. Since additively homomorphic encryption is like addition operation [25], linearity is preserved such that $\langle \boldsymbol{x} \rangle + \langle \boldsymbol{y} \rangle = \langle \boldsymbol{x} + \boldsymbol{y} \rangle$. Through applying additively homomorphic encryption, model parameters can be shared without leaking users' private information. Other encryption approaches can also be used in this FL procedure, which can be studied in the future.

In our framework, we exploit a lightweight model named GhostNet [33], which is quite suitable for edge devices because of its low computational cost and inference latency. The learning goal for each target device $\boldsymbol{p}$'s model is represented as

$$\arg\min_{\Theta^{\boldsymbol{p}}} \mathcal{L}_{\boldsymbol{p}} = \sum_{i=1}^{n^{\boldsymbol{p}}} \mathcal{C}\left(\boldsymbol{l}_i^{\boldsymbol{p}}, \boldsymbol{M}_{\boldsymbol{p}}\left(\boldsymbol{X}_i^{\boldsymbol{p}}\right)\right) \tag{3}$$

where $\Theta^{\boldsymbol{p}}$ means the optimal weights and biases of all layers for $\boldsymbol{p}$'s model, $\mathcal{C}(\cdot,\cdot)$ denotes as the classification loss function for model training, i.e., cross-entropy loss function, $\mathcal{L}_{\boldsymbol{p}}$ means the overall loss for model training and $\boldsymbol{M}_{\boldsymbol{p}}$ represents $\boldsymbol{p}$'s model. $\{\boldsymbol{X}_i^{\boldsymbol{p}},\ \boldsymbol{l}_i^{\boldsymbol{p}}\}_{i=1}^{n^{\boldsymbol{p}}}$ are samples and their corresponding true labels from $\boldsymbol{p}$'s dataset with size $\boldsymbol{n}^{\boldsymbol{p}}$.

When all users' models $\boldsymbol{M}_{\boldsymbol{p}}$ are trained after certain epochs during one communication round, non-classification layer parameters of these models are uploaded to the cloud server for updating the cloud feature extractor. As stated in [34], it has been evaluated that a high-quality cloud model can be trained with relatively few communication rounds through the federated averaging method, which averages parameters of all user models in the cloud server to update the cloud model. Hence, we exploit this method as our FL method. Assume that non-classification layer parameters of $\boldsymbol{N}$ user models are uploaded to the cloud server for updating the cloud feature extractor, then the parameters $\boldsymbol{\Theta_c}$ of cloud feature extractor $\boldsymbol{M_c}$ after updating is represented as

$$\boldsymbol{\Theta_c} = \frac{1}{\boldsymbol{N}} \sum_{\boldsymbol{n}=1}^{\boldsymbol{N}} \boldsymbol{\Theta}_{\boldsymbol{l}}^{\boldsymbol{p_n}} \tag{4}$$

where we denote $\boldsymbol{\Theta}_{\boldsymbol{l}}^{\boldsymbol{p_n}}$ as the non-classification layer parameters of a user model, and we average the non-classification layer parameters of user models to update the cloud feature extractor here. In the future, we will further study the impact of specific model parameters during averaging. The updated cloud feature extractor $\boldsymbol{M_c}$ has a robust feature extraction performance after sufficient rounds of communication between users and the cloud server. New users can also take part in the subsequent training rounds to help train the cloud feature extractor. Therefore, our proposed framework is highly extensible and can be constantly updated.

FL is applied to tackle the data island problem, while the personalized training problem still exists. Hence, transfer learning is adopted in our proposed framework to address this issue. As mentioned before, the front layers of a neural network are highly reusable. They usually extract general features, and subsequent layers extract more specific features for the task. Therefore, after combining the global cloud feature extractor and a user's classifier, we apply deep transfer learning to train a personalized model for a new user. Fig. 3 shows the model transfer procedure in our proposed framework and the details of the Ghost Bottleneck and Ghost Module. Note that we also update those Batch Normalization layer parameters in the frozen part since they are more specific to the local training data. The major component of our model is the Ghost Module, which mainly uses cheap linear operation to augment features and generate more feature maps after obtaining a few inherent channels by a regular convolutional operation [33]. The Ghost bottleneck with stride=1 comprises two Ghost Modules, which that with stride=2 further includes a depthwise convolutional with stride=2 between the two Ghost Modules. The SE Module, which can utilize the correlation between feature maps, is also adopted in some Ghost bottlenecks, as shown in Fig. 3. For details of identifying the critical layers for the model that can be transferred, the reader is referred to the discussion in Section V.

C. Online Location Estimation Stage

In the offline training phase, the number of neurons at the model output layer is equal to the number of RPs for training. Thus, each neuron output corresponds to an RP. According to [13], probabilistic methods perform better than deterministic methods. Hence, we use the softmax activation function for the model output, which maps the outputs to the range [0, 1] to estimate the posterior probabilities output. Each neuron output after the softmax activation function is given by

$$P(L_i|v_{test}) = \frac{e^{z_i}}{\sum_{j=1}^{N} e^{z_j}}. \tag{5}$$

In (4), z_i and z_j are the neuron outputs with corresponding RPs i and j, L_i is the event that the node is located at RP i, v_{test}

Algorithm 1 Probabilistic-based position estimation

Input: model output o

1: Compute posterior probabilities $\{P(L_i|v_{test})\}_{i=1}^{N}$ of o
2: Obtain $\{(x_i, y_i)\}_{i=1}^{k}$ which have k largest posterior probabilities $\{P(L_i|v_{test})\}_{i=1}^{k}$
3: Obtain (x_i, y_i) which has maximum posterior probability $P_{max}(L_i|v_{test})$
4: $t \leftarrow$ predefined threshold
5: **if** $P_{max}(L_i|v_{test}) > t$ **then**
6: $\hat{L} = (x_i, y_i)$
7: **else**
8: $\hat{L} = \left(\frac{\sum_{i=1}^{k} x_i}{k}, \frac{\sum_{i=1}^{k} y_i}{k}\right)$
9: **end if**

Output: estimated position $\hat{L}$

is the input test data, $P(L_i|v_{test})$ is the posterior probability and N is the total number of RPs in the radio-map.

In the online location estimation stage, the CSI radio image of the mobile device is fed into the model. The neuron output o_i can be interpreted as the posterior probability that the mobile device is located at RP i. Since the mobile device could appear in any position within the interesting area, we combine the maximum posterior method and average posterior method based on a predefined threshold. Assume that the positions of k RPs which have largest posterior probabilities are $\{(x_i, y_i)\}_{i=1}^{k}$. Formulas for these two methods are shown in (6) and (7) respectively.

$$\hat{L} = \arg\max_{L_i}\{P(L_i|v_{test})\}_{i=1}^{N} \tag{6}$$

$$\hat{L} = \left(\frac{\sum_{i=1}^{k} x_i}{k}, \frac{\sum_{i=1}^{k} y_i}{k}\right) \tag{7}$$

where $\hat{L}$ is the estimated position and only k RPs with the largest posterior probabilities are considered in the average position estimation method in (7). The pseudocode for online location estimation is presented in Algorithm 1. The input to the algorithm is the model output o after inferring the i-th input CSI image. First, we compute the posterior probability of the model output through the softmax activation function. Then we identify k reference locations $\{(x_i, y_i)\}_{i=1}^{k}$ which have the largest posterior probabilities based on the topk function (i.e., obtain K largest elements of the given input data along a given dimension) and the reference location L_i with the maximum posterior probability. Since the mobile device could be located at fingerprinted reference locations, we set a threshold t. If $P(L_i|v_{test}) < t$, we will average $\{(x_i, y_i)\}_{i=1}^{k}$ in (7) to estimate the current position as illustrated in Algorithm 1.

IV. Experiment Setup

A. Equipment and Data Collection Method

Fig. 4 illustrates the communication between the devices for CSI data collection. The PC is used to send control messages to the AP for generating the wireless signals that contain CSI data, while a Raspberry Pi 4B serves as a data collector to extract CSI data at different positions. Ping packets are sent from the PC to the AP, and pong packets are sent back from the AP to the PC.

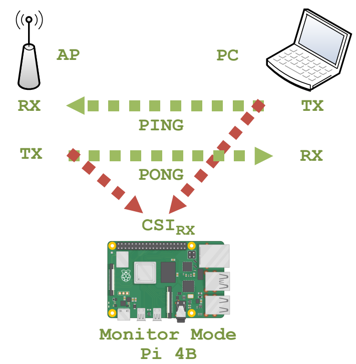


Fig. 4: CSI data collection method.

The Pi 4B in monitor mode was configured with Raspberry Pi OS (Buster/Linux 5.4.83) with the main branch of nexmon_csi[1] installed. The following filter parameters were used for configuring nexmon: channel 36/80, Core 1, and NSS mask 1. Each Raspberry Pi 4B was paired to the AP. To reduce interference, a computer was connected to the Pi 4B over the SSH protocol on a separate 2.4 GHz channel to control the data collection. The 5 GHz channel of the AP operates on channel 36 with a bandwidth of 80 MHz. Note that the model of the AP is not restricted as long as it supports the 802.11ac protocol. Finally, the PC is connected to the 5 GHz channel of the AP to generate data flow from which the Pi 4B in monitor mode can capture CSI data. The PC is configured to send ping flood to the AP at a rate of 1000 Hz.

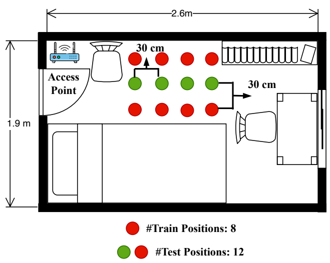
B. Environments and Datasets

TABLE I: CSI Data Collection Environments

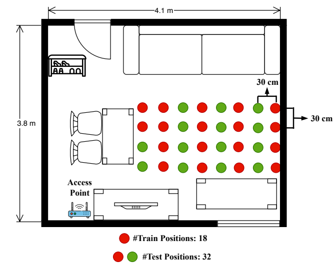
Index	Environments	#Fingerprints	#Positions	#Images
A	Harbour Place A	37	64	21,536
B	Harbour Place B	18	32	22,016
C	Harbour Place C	8	12	8,960
D	City One A	8	12	8,960
E	City One B	8	12	8,960
F	City One C	8	12	8,960
G	PolyU Office	8	12	8,960
H	HaoJiang	122	238	34,114

Eight different environments were considered for CSI data collection. Fig. 5 show the layout of Environments A, B, E and H with 64, 32, 12 and 238 training/testing positions, respectively. The training/testing position layout of other 12-position environments is similar to Environment E. In each environment, we used the nexmon CSI patch to collect training/testing positions that are 30 cm apart. The tcpdump command is used to render CSI data for producing a pcap file. An open-source software named csiread is used for interpreting this file, which generates a 256 x 1000 CSI amplitude matrix after computing the absolute values and the transpose operation. Note that 256 means the number of sub-carriers, and 1000 represents

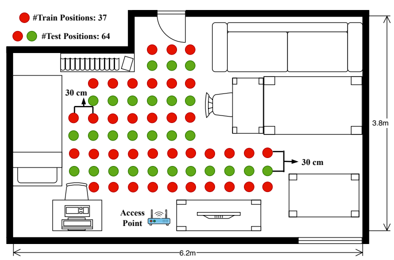
[1]https://github.com/seemoo-lab/nexmon



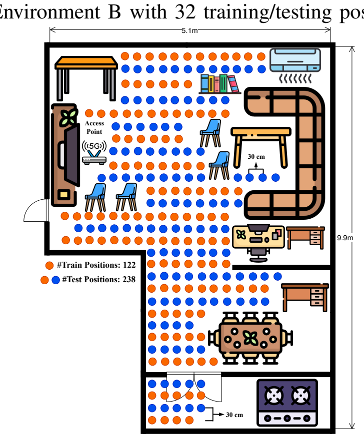
(a) Environment E with 12 training/testing positions.



(b) Environment B with 32 training/testing positions.



(c) Environment A with 64 training/testing positions.



(d) Environment H with 238 training/testing positions.

Fig. 5: Different environments with respect to four different kind of training/testing position layouts.

the number of received data packets. Then we further slice the matrix into four sub-matrices as 256 x 250 and save them as radio images with 224 x 224 resolution after data preprocessing for model training. Table I lists the names of our CSI data collection environments, their corresponding indexes, the number of fingerprints that are used in the training phase, the total number of training/testing positions, and the size of the dataset in each environment.

C. Data Preprocessing

Since signal preprocessing or filtering can degrade the timeliness of a real-time system, we minimize the signal preprocessing in our system such that raw CSI data is used with 8 Pilot sub-carriers and 14 Null sub-carriers (including Guard sub-carriers) filtered away before saving them as CSI radio images according to [35], [36].

D. Model Training Configuration

Our experiments exploit the GhostNet model, which is suitable for building a real-time indoor localization system because of its high computational efficiency and performance. The critical architecture of this model is the Ghost module, which produces more feature maps with less computational cost. Details of it can be found in [33]. Our model is implemented by Pytorch and trained using a computer configured with Nvidia RTX 3090 GPU. Before training the model, we use some data augmentation methods[2] (i.e., RandomRiszedCrop, RandomHorizontalFlip, and ColorJitter) when loading our training data. The AdamW [37] optimizer is used to optimize model parameters during the training phase. StepLR scheduler is also applied to attenuate the learning rate periodically. Cross-entropy loss function with label smoothing [38] is adopted. Each dataset is roughly divided into two sub-datasets, i.e., a training dataset, which is 80% of it, and a testing dataset, which is 20% of it. The batch size at each epoch is set to 64. For the threshold t and the number of RPs k that are used for averaging in Algorithm 1, we set them to 0.85 and seven respectively in the experiment.

In our experiments, we divide our model training into three parts: (i) We use the dataset of Environments A, B and H which has 64 positions, 32 positions and 238 positions respectively to train the GhostNet model for evaluating our proposed location estimation method and compare it with three other existing CSI-based position estimation algorithms; (ii) We apply FL, which averages non-classification layer parameters of trained user models in different combinations of six environments to build a cloud feature extractor. In this part, we set the iterative rounds to 300 with one local training epoch, and; (iii) We utilize a deep transfer learning method (i.e., fine-tune) to determine the number of layers that are important for extracting common features (i.e., need to be reused) in a new environment and evaluate the federated transferability. For normal and deep transfer training, we set the training epochs to 80. For determining the number of reused layers, we further apply the early stop method in deep transfer training to halt the training process as long as there is no more localization accuracy improvement after ten training epochs. We record the mean localization error δ as localization accuracy. Assume that the estimated position of an unknown position i is $(\hat{x_i}, \hat{y_i})$ and

[2]https://pytorch.org/vision/stable/transforms.html

the ground truth position of that is (x_i, y_i). For N locations, the mean localization error is computed as shown in (8).

$$\delta = \frac{\sum_{i=1}^{N} \sqrt{(\hat{x_i} - x_i)^2 + (\hat{y_i} - y_i)^2}}{N} \tag{8}$$

V. Experiment Evaluation

In this section, we use CSI packets collected by the Raspberry Pi 4B platform via Wi-Fi channels from a single router as positioning features to validate our proposed position estimation scheme in Environments A and B. We also identify the number of critical layers that are important for extracting common features in the cloud feature extractors. Finally, the performance of the proposed FedPos framework is evaluated by comparing it with normal training, federated learning only, and conventional ML approaches.

A. Localization Performance in Real-world Environments

We first evaluate the localization performance of our proposed position estimation method. For comparison, we reproduced the results of seven existing traditional and learning-based methods, including TRRS [10], TRRS with CFA, WFA and FA [11], CSI-MIMO [12], DeepFi [13] and ConFi [4]. In CSI-MIMO, RP with the maximum posterior probability is selected as the estimated position. The formulation for this method can be seen in (6). DeepFi uses the weighted average of all RPs as the estimated position and ConFi adopts the weighted centroid method to estimate the position.

TABLE II: Mean localization errors (cm) for Environments A, B and H. Environment is denoted as Env.

Method	Env A	Env B	Env H
DeeFi	50.65	66.75	73.56
ConFi	51.20	71.62	65.89
CSI-MIMO	58.05	76.12	74.87
TRRS	153.81	116.63	344.70
TRRS with CFA	128.27	98.26	329.99
TRRS with FA	136.56	112.63	293.26
TRRS with WFA	135.45	98.57	303.14
Our Proposed Method	**43.13**	**55.61**	**63.58**

We use CSI data collected from Environments A, B and H as shown in Fig 5 (b), (c) and (d) to evaluate our proposed position estimation method. Table II shows the localization accuracy of each environment, and Table III describes the mean GPU run time of Environment A and B. In Environment A, the mean localization error of the proposed method with a single router is about 43.13 cm. In Environment B, where the room size is smaller than A and there are more objects which results in a more severe multi-path and shadowing effect, the mean localization error is about 55.61 cm across 32 testing locations. In Environment H, where the room size is of a much larger scale than those in previous experiments and with much more training/testing locations, the mean localization error is about 63.58 cm across 238 testing positions. Our proposed method outperforms seven other existing CSI-based position estimation algorithms in this scenario. Our method achieves a 14.85% localization performance boost when compares with DeepFi which achieves the second smallest mean localization error (i.e., 50.65 cm) in Environment A. For Environment B, we also obtain a 16.69% performance improvement over DeepFi which achieves the second smallest mean localization error (i.e., 66.75 cm).

According to Table II, we can see that all the learning-based methods perform better than the other four traditional methods, which indicates the great advantage of using learning-based approaches in CSI-based indoor positioning. Since the TRRS method does not leverage historical fingerprint data, it has the worst performance among the four traditional methods. Among these four traditional methods, TRRS with CFA has the best localization performance in the two environments. It is because this method requires all fingerprints to be stored in the database. One of those fingerprints in the database is likely to be a replica of the fingerprint currently being tested for localization. In addition, it requires a relatively large database storage capacity, and the processing time may be longer than other methods since it requires computing the TRRS for every fingerprint. Compared with TRRS with CFA, machine learning-based approaches only need to store the model parameters in the local device rather than a huge amount of historical fingerprint data, hence, saving the storage capacity. For TRRS with FA and TRRS with WFA, although they average all historical fingerprints to leverage all the data features and reduce the need to store a large number of fingerprints, the averaged fingerprint is more susceptible to outliers and may decrease the quality of data. Therefore, the localization performance of these methods is relatively poor.

For the inference time, CSI-MIMO only considers fingerprinted RPs, it achieves the worst localization performance. For a mean run time, CSI-MIMO confirms to be the fastest one in both environments since it only needs to search the RP with the maximum posterior probability. Therefore, the computational cost is lower than the other three methods. However, our proposed method ensures the best localization accuracy without compromising the GPU run time when compared with the other three approaches.

TABLE III: Mean GPU run times (ms) for Environments A and B

Method	Environment A	Environment B
DeeFi	19.12	18.85
ConFi	19.51	18.41
CSI-MIMO	**18.78**	**18.31**
Our Proposed Method	19.17	18.50

Fig. 6 shows the cumulative distribution function (CDF) of distance errors of the eight methods in Environment A. This environment includes LOS and NLOS scenarios. For the proposed method, about 85% of all testing points have a localization error under 100 cm. DeepFi, on the other hand, ensures that about 80% of all testing points have a localization error under 100 cm. Moreover, about 47% of all testing points for all learning-based methods have no distance error.

Fig. 7 shows the CDF of distance errors of the eight methods in Environment B. This environment is an NLOS scenario. In this more complicated propagation environment, our method confirms that over 95% of all testing points have a distance

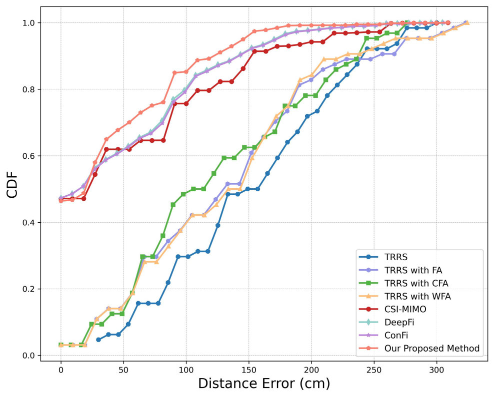


Fig. 6: CDF of localization errors in Environment A.

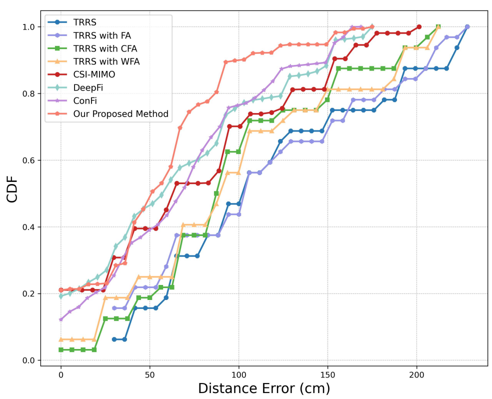


Fig. 7: CDF of localization errors in Environment B.

error under 125 cm, which is the most accurate among the four. On the other hand, DeepFi ensures that about 80% of all testing points have a distance error under 125 cm. In addition, the correlation between signal strength and propagation distance is weaker than that in Environment A since the cabinet, television and tables obstruct almost all LOS paths and result in a more significant multi-path and shadowing effect. The other three methods all perform worse than that in Environment A. It is also noticed in Fig. 7 that about 90% of all testing points have a localization error under 125 cm for ConFi and only 75% for CSI-MIMO. Although the result shown in Fig. 7 indicates that our method has huge advantages over other methods, there is poor performance where the CDF is close to 1. According to our observation, we believe that it is because our method mainly estimates the position by averaging the seven possible reference positions with the top seven largest posterior probabilities. And some testing points are far away from some of these possible reference positions, resulting in larger estimation errors. Therefore, the localization errors for this set of testing points are larger than other estimation cases and result in poor performance when the CDF is close to 1.

Fig. 8 describes the CDF of distance errors of the eight methods in Environment H. This environment is a relatively larger environment (50.49 m^2) with 238 training/testing positions, which is of a much larger scale than those in previous experiments. This environment also considers both LOS and NLOS scenarios. We evaluate our position estimation method

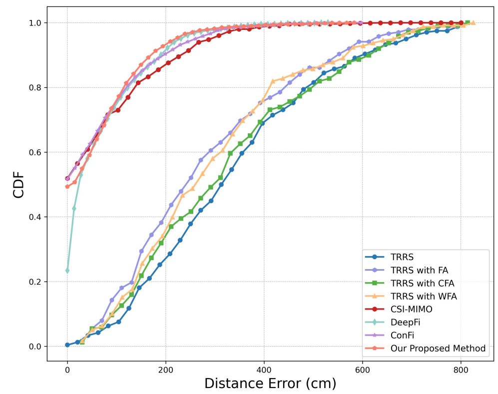


Fig. 8: CDF of localization errors in Environment H.

and compare it with the other learning-based and traditional methods. The experimental results demonstrate that all the machine learning-based methods have a huge advantage when compared with the other four traditional approaches. According to the figure shown above, our proposed method still has a slight performance boost when compared with other learning-based methods. In Environment H, the localization performance of TRRS with FA and WFA outperforms that of TRRS with CFA. This is because the number of outliers in this environment is less than those in other environments, and the averaged fingerprint data leverage all the data features and reduces the impact of random factors.

B. Federated Transferability

In this part, we conduct extensive experiments to evaluate the performance of FedPos. Fig. 2 shows the overview of our proposed framework, and Fig. 3 illustrates the transfer learning procedure. Our GhostNet model comprises 16 Ghost Bottleneck blocks (each contains four to five convolutional layers), three convolutional layers and a fully connected layer with only about 3.95 million parameters. First, we apply FL for different combinations of six environments to build the corresponding cloud feature extractors which have 16 Ghost bottleneck blocks and three convolutional layers. We regard the Ghost bottleneck block as one layer when conducting transfer learning. Hence, there are a total of 19 layers for reusing. Then we gradually increase the number of layers in the cloud feature extractors to be reused to train the new user model, from the first layer to all layers. In this process, when we feed our CSI data collected from new environments to the new user model, Fig. 9, 10 and 11 show the change of the corresponding localization accuracy and training time. For 12-position environments (Env C to G), we average their changes to form the overall trend in Fig. 11. Qualitatively, it can be seen that localization accuracy and training time gradually decrease when the number of reusing layers increases. As shown in Fig. 9 and 11, distance errors increase dramatically after reusing over four layers, and turning points of training time at layer eleventh and layer ninth can be observed respectively. In Fig. 10, a turning point of localization error at layer sixth can be observed, whereas the training time is greatly reduced at layer 13th. According to these results, we

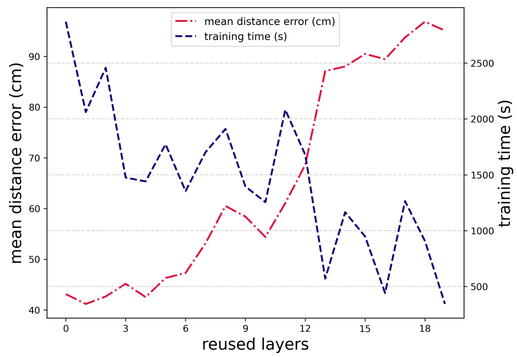


Fig. 9: The performance change of the model when transferring to Environment A layer by layer.

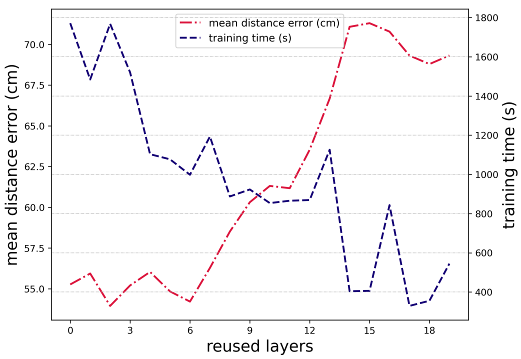


Fig. 10: The performance change of the model when transferring to Environment B layer by layer.

can say that for a pre-trained 19 Ghost bottlenecks cloud feature extractor for indoor positioning in all environments except Environment B, its first four layers can extract common features in CSI data, and hence they can be reused in other models without retraining or tuning, whereas the subsequent 15-layer parameters still need to be fine-tuned, since they may focus more on environment-specific features. For Environment B, the first six layers of the cloud feature extractor can be reused, while its subsequent 13-layer parameters still need to be fine-tuned. We set the number of reused layers according to the results above in the deep transfer training for the following experiments for comparison.

The deep transfer learning approach as introduced in Fig. 3 is adopted to conduct personalized training for each environment. For brevity in notation, we use NoFed and FedOnly to denote the normal and federated training setting. We can see that the localization performance of FedOnly for each environment is worse than NoFed and FedPos since the cloud feature extractor is general and only contains coarse information from other clients. This result indicates that after introducing the fine-tuning method, the personalization problem can be minimized. We also compare the localization performance of FedPos with KNN and random forest (RF) which use probability as the final output. Table IV shows the comparison results of mean distance errors for different model training methods in a new environment. We can see that our proposed FedPos framework ensures the lowest localization errors in all environments. Compared with NoFed, it has a

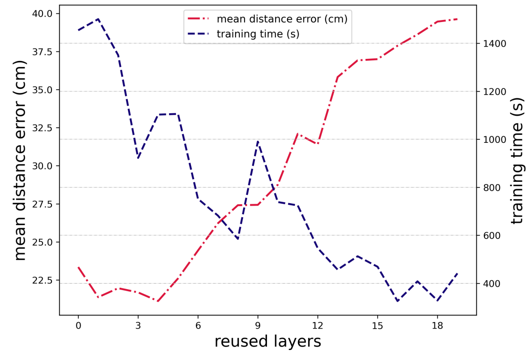


Fig. 11: The average performance change when transfer to Environments C, D, E, F, and G layer by layer.

TABLE IV: The comparison results of mean localization errors (cm) for models trained using traditional ML methods, randomly initialized weights, federated learning only, and federated transfer learning in new environments

New Environments	KNN	RF	NoFed	FedOnly	FedPos
A	89.85	77.59	43.13	86.02	**42.18**
B	66.25	74.91	55.61	65.32	**54.55**
C	40.57	38.71	23.68	32.34	**20.01**
D	39.47	39.47	23.33	36.60	**20.90**
E	37.73	38.24	22.28	30.83	**21.43**
F	35.84	36.32	23.75	37.59	**22.59**
G	39.29	40.21	21.46	39.76	**20.44**
AVG	49.86	49.35	30.46	46.92	**28.87**

5.22% average performance boost. FedPos also confirms lower distance errors when compared with conventional ML methods (e.g., KNN, RF). In conclusion, it proves the effectiveness of our proposed FedPos framework. On the other hand, we can also observe that for indoor positioning, DL methods (NoFed, FedPos) ensure lower distance errors than conventional ML methods (KNN, RF). This is because of the strong representation capability of deep CNN, whereas hand-crafted feature learning is required in conventional ML methods. Another benefit of DL is that it allows the model to be updated online incrementally without retraining from the beginning, whereas conventional methods require additional incremental algorithms. This property is particularly useful in federated transfer learning, where model reuse is critical.

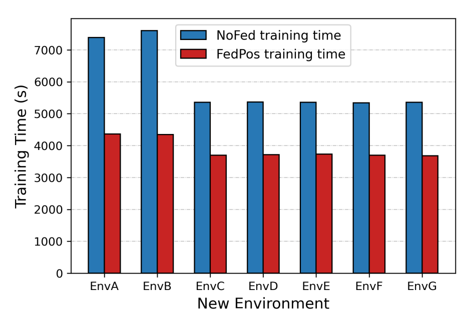


Fig. 12: The comparison result of training time for NoFed and FedPos. Environment is denoted as Env.

We also compare the training time of our proposed FedPos framework with NoFed. Fig. 12 shows the comparison result of training time for NoFed and our proposed FedPos framework. We can still clearly see that the training time of FedPos is faster than that of NoFed for each new environment, with a saving of about 34.78% averaging training time. This result shows that the superior performance of FedPos in localization does not come along with any sacrifice in training time. After federated training, the front layers are sufficiently robust to be reused by other datasets. It enhances the localization performance without compromising the training time.

C. Impact of training dataset dropout ratio

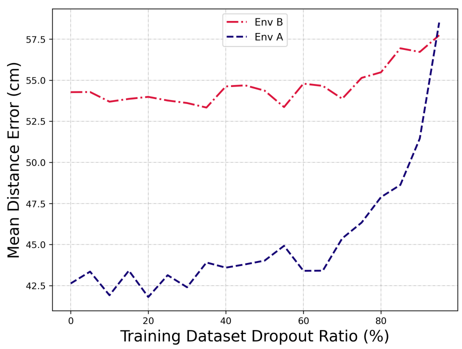


Fig. 13: The transfer training results in Environments A and B when a certain percentage of training data is randomly dropped.

In this section, we further study the impact of training dataset size on model retraining in Environments A and B since transfer learning is a useful approach to deal with the problem of insufficient training data in ML [39]. For Environment A, we reuse the first four layers of the cloud encoder, whereas, in Environment B, the first five layers of the cloud encoder are transferred. We randomly drop a certain percentage of training data in the training dataset to improve the retraining efficiency of the model. Fig 13 shows the localization results, where the range for the proportion of training data dropout is set to [0%, 95%] with a 5% interval. It can be noticed that a similar localization error can still be obtained when using 35% of training data for transfer training in Environment A, whereas only 30% of training data are required for transfer learning to achieve a similar localization performance in Environment B. That may be because the number of reused cloud encoder layers in Environment B is more than that in Environment A. Therefore, the advantage of transfer learning can be better leveraged. Overall, it can be summarized that by utilizing transfer learning and part of the layers in the cloud encoder, at least 65% of training data can be saved to achieve a similar level of localization performance.

VI. Conclusion and Future Works

This study proposed FedPos, a federated transfer learning framework together with a new position estimation method that exploits CSI as the localization feature for Wi-Fi indoor positioning. The proposed FedPos framework aggregates non-classification layer parameters of models trained from different environments to build a cloud feature extractor while preserving privacy. It constructs personalized models through fine-tuning and tackles privacy leakage and personalized training problems. We have evaluated the localization performance using the Raspberry Pi 4B platform. Our experimental results show that our proposed position estimation method can guarantee lower distance errors than three other existing CSI-learning-based position estimation algorithms, with up to 16.69% performance improvement over DeepFi [13]. Our results also indicate that the proposed FedPos framework has strong transferability, which achieves a 5.22% average localization performance boost with only requiring 65.22% average training time when compared with normal training.

Although only a small amount of devices support extracting CSI data from off-the-shelf routers currently, we believe that an increasing number of chip manufacturers will open up the CSI extraction functionalities with the fast development and growing demand for Wi-Fi sensing technologies. Hence, CSI data are expected to be crowdsourced in the near future like GPS data nowadays. The proposed framework in this paper will further benefit from this development and demonstrate a significant advantage in indoor localization tasks. For future research, we plan to investigate adaptive fine-tuning using a policy network to further improve the efficiency and performance of transfer training as well as how to collect CSI data with the assistance of robots to automate the site survey process. The impact of different combinations for parameters k and t in our proposed position estimation method and how to dynamically adjust this parameter combination according to the environment will also be further investigated in the future.

References

[1] Y. Xu, *Autonomous Indoor Localization Using Unsupervised Wi-Fi Fingerprinting*. kassel university press GmbH, 2016.

[2] J. M. Rocamora, I. W.-H. Ho, W.-M. Mak, and A. P.-T. Lau, "Survey of CSI fingerprinting-based indoor positioning and mobility tracking systems," *IET Signal Processing*, vol. 14, no. 7, pp. 407–419, 2020.

[3] C. Liu, D. Fang, Z. Yang, H. Jiang, X. Chen, W. Wang, T. Xing, and L. Cai, "RSS distribution-based passive localization and its application in sensor networks," *IEEE Transactions on Wireless Communications*, vol. 15, no. 4, pp. 2883–2895, Apr. 2016.

[4] H. Chen, Y. Zhang, W. Li, X. Tao, and P. Zhang, "ConFi: Convolutional neural networks based indoor wi-fi localization using channel state information," *IEEE Access*, vol. 5, pp. 18 066–18 074, 2017.

[5] W. Xue, X. Hua, Q. Li, K. Yu, W. Qiu, B. Zhou, and K. Cheng, "A new weighted algorithm based on the uneven spatial resolution of RSSI for indoor localization," *IEEE Access*, vol. 6, pp. 26 588–26 595, 2018.

[6] M. T. Hoang, B. Yuen, X. Dong, T. Lu, R. Westendorp, and K. Reddy, "Recurrent neural networks for accurate RSSI indoor localization," *IEEE Internet of Things Journal*, vol. 6, no. 6, pp. 10 639–10 651, Dec. 2019.

[7] M. T. Hoang, B. Yuen, K. Ren, X. Dong, T. Lu, R. Westendorp, and K. Reddy, "A cnn-lstm quantifier for single access point csi indoor localization," 2020. [Online]. Available: https://arxiv.org/abs/2005.06394

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

[9] D. Halperin, W. Hu, A. Sheth, and D. Wetherall, "Tool release: gathering 802.11n traces with channel state information," *ACM SIGCOMM Computer Communication Review*, vol. 41, no. 1, pp. 53–53, Jan. 2011.

[10] C. Chen, Y. Chen, Y. Han, H.-Q. Lai, and K. J. R. Liu, "Achieving centimeter-accuracy indoor localization on wifi platforms: A frequency hopping approach," *IEEE Internet of Things Journal*, vol. 4, no. 1, pp. 111–121, 2017.

[11] E. R. Magsino, I. W.-H. Ho, and Z. Situ, "The effects of dynamic environment on channel frequency response-based indoor positioning," in *2017 IEEE 28th Annual International Symposium on Personal, Indoor, and Mobile Radio Communications (PIMRC)*, 2017, pp. 1–6.

[12] Y. Chapre, A. Ignjatovic, A. Seneviratne, and S. Jha, "CSI-MIMO: Indoor wi-fi fingerprinting system," in *39th Annual IEEE Conference on Local Computer Networks*, 2014.

[13] X. Wang, L. Gao, S. Mao, and S. Pandey, "CSI-based fingerprinting for indoor localization: A deep learning approach," *IEEE Transactions on Vehicular Technology*, vol. 66, no. 1, pp. 763–776, 2017.

[14] F. Gringoli, M. Schulz, J. Link, and M. Hollick, "Free your CSI: A channel state information extraction platform for modern wi-fi chipsets," in *Proceedings of the 13th International Workshop on Wireless Network Testbeds, Experimental Evaluation & Characterization - WiNTECH '19*. ACM Press, 2019.

[15] X. Wang, L. Gao, and S. Mao, "BiLoc: Bi-modal deep learning for indoor localization with commodity 5ghz WiFi," *IEEE Access*, vol. 5, pp. 4209–4220, 2017.

[16] P. Voigt and A. Von dem Bussche, "The eu general data protection regulation (gdpr)," *A Practical Guide, 1st Ed., Cham: Springer International Publishing*, vol. 10, no. 3152676, pp. 10–5555, 2017.

[17] N. Inkster, *China's cyber power*. Routledge, 2018.

[18] Q. Yang, Y. Liu, T. Chen, and Y. Tong, "Federated machine learning: Concept and applications," *ACM Transactions on Intelligent Systems and Technology*, vol. 10, no. 2, pp. 1–19, Mar. 2019.

[19] D. C. Nguyen, M. Ding, P. N. Pathirana, A. Seneviratne, J. Li, and H. V. Poor, "Federated learning for internet of things: A comprehensive survey," *IEEE Communications Surveys & Tutorials*, vol. 23, no. 3, pp. 1622–1658, 2021.

[20] Y. Liu, H. Li, J. Xiao, and H. Jin, "FLoc: Fingerprint-based indoor localization system under a federated learning updating framework," in *2019 15th International Conference on Mobile Ad-Hoc and Sensor Networks (MSN)*. IEEE, Dec. 2019.

[21] B. S. Ciftler, A. Albaseer, N. Lasla, and M. Abdallah, "Federated learning for RSS fingerprint-based localization: A privacy-preserving crowdsourcing method," in *2020 International Wireless Communications and Mobile Computing (IWCMC)*. IEEE, Jun. 2020.

[22] H. Zhu, J. Xu, S. Liu, and Y. Jin, "Federated learning on non-IID data: A survey," *Neurocomputing*, vol. 465, pp. 371–390, Nov. 2021.

[23] R. L. Rivest, L. Adleman, M. L. Dertouzos *et al.*, "On data banks and privacy homomorphisms," *Foundations of secure computation*, vol. 4, no. 11, pp. 169–180, 1978.

[24] Y. Chen, X. Qin, J. Wang, C. Yu, and W. Gao, "FedHealth: A federated transfer learning framework for wearable healthcare," *IEEE Intelligent Systems*, vol. 35, no. 4, pp. 83–93, Jul. 2020.

[25] H. Yang, H. He, W. Zhang, and X. Cao, "FedSteg: A federated transfer learning framework for secure image steganalysis," *IEEE Transactions on Network Science and Engineering*, vol. 8, no. 2, pp. 1084–1094, Apr. 2021.

[26] W. Liu, Q. Cheng, Z. Deng, H. Chen, X. Fu, X. Zheng, S. Zheng, C. Chen, and S. Wang, "Survey on CSI-based indoor positioning systems and recent advances," in *2019 International Conference on Indoor Positioning and Indoor Navigation (IPIN)*. IEEE, 2019.

[27] G. Forbes, S. Massie, and S. Craw, "WiFi-based human activity recognition using raspberry pi," in *2020 IEEE 32nd International Conference on Tools with Artificial Intelligence (ICTAI)*. IEEE, Nov. 2020.

[28] J. Wang, L. Zhang, Q. Gao, M. Pan, and H. Wang, "Device-free wireless sensing in complex scenarios using spatial structural information," *IEEE Transactions on Wireless Communications*, vol. 17, no. 4, pp. 2432–2442, Apr. 2018.

[29] Q. Gao, J. Wang, X. Ma, X. Feng, and H. Wang, "CSI-based device-free wireless localization and activity recognition using radio image features," *IEEE Transactions on Vehicular Technology*, vol. 66, no. 11, pp. 10 346–10 356, Nov. 2017.

[30] D. Khan and I. W.-H. Ho, "Crosscount: Efficient device-free crowd counting by leveraging transfer learning," *IEEE Internet of Things Journal*, pp. 1–1, 2022.

[31] S. J. Pan and Q. Yang, "A survey on transfer learning," *IEEE Transactions on Knowledge and Data Engineering*, vol. 22, no. 10, pp. 1345–1359, Oct. 2010.

[32] P. Li, H. Cui, A. Khan, U. Raza, R. Piechocki, A. Doufexi, and T. Farnham, "Deep transfer learning for WiFi localization," in *2021 IEEE Radar Conference (RadarConf21)*. IEEE, May 2021.

[33] K. Han, Y. Wang, Q. Tian, J. Guo, C. Xu, and C. Xu, "GhostNet: More features from cheap operations," in *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, Jun. 2020.

[34] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, "Communication-efficient learning of deep networks from decentralized data," *Artificial intelligence and statistics*, pp. 1273–1282, 2017.

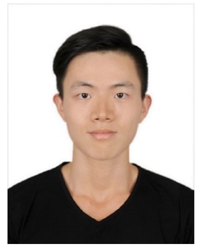
[35] A. R. Voggu, V. Vazhayily, and M. Ra, "Decimeter level indoor localisation with a single WiFi router using CSI fingerprinting," in *2021 IEEE Wireless Communications and Networking Conference (WCNC)*. IEEE, Mar. 2021.

[36] M. S. Gast, *802.11 ac: a survival guide: Wi-Fi at gigabit and beyond.* " O'Reilly Media, Inc.", 2013.

[37] I. Loshchilov and F. Hutter, "Decoupled weight decay regularization," 2017. [Online]. Available: https://arxiv.org/abs/1711.05101

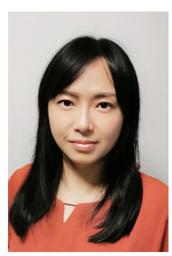
[38] R. Müller, S. Kornblith, and G. E. Hinton, "When does label smoothing help?" *Advances in neural information processing systems*, vol. 32, 2019.

[39] C. Tan, F. Sun, T. Kong, W. Zhang, C. Yang, and C. Liu, "A survey on deep transfer learning," *International conference on artificial neural networks*, pp. 270–279, 2018.



Jingtao Guo received the B.Eng. degree from Huizhou University, China, and the M.Sc. degree in electronic and information engineering from The Hong Kong Polytechnic University, Hong Kong, in 2020 and 2022, respectively. He is currently a Research Assistant with the Department of Electronic and Information Engineering, The Hong Kong Polytechnic University, Hong Kong. His research interests include wireless sensing, federated learning, and transfer learning. His research focuses mainly on personalized federated learning for large-scale wireless sensing applications.

Ivan Wang-Hei Ho (M'10–SM'18) received the B.Eng. and M.Phil. degrees in information engineering from The Chinese University of Hong Kong, Hong Kong, in 2004 and 2006, respectively, and the Ph.D. degree in electrical and electronic engineering from Imperial College London, U.K., in 2010. In 2007, Ivan spent a summer working at the IBM T. J. Watson Research Center, USA. After his Ph.D. graduation, he was with the System Engineering Initiative at Imperial College as a Postdoctoral Research Associate. In Sept 2010, he co-founded P2 Mobile Technologies Limited and served as the Chief Engineer. He primarily invented the MeshRanger series wireless mesh embedded system, which received the Silver Award in Best Ubiquitous Networking at the Hong Kong ICT Awards 2012. He is currently an Associate Professor with the Department of Electronic and Information Engineering, The Hong Kong Polytechnic University, Hong Kong. His research interests include wireless communications and networking, specifically in vehicular networks, intelligent transportation systems (ITS), and Internet of things (IoT). His work on indoor positioning and IoT received a number of awards, including the Gold Medal at iENA 2019, the Gold Medal with the Organizer's Choice Award at iCAN 2020, and the Gold Medal at the International Exhibition of Inventions Geneva in 2021.



Yun Hou (SM'20) received her MSc and PhD degrees in Electrical and Electronic Engineering from Imperial College London, UK, in 2006 and 2009 respectively. She is currently an Assistant Professor with the Department of Computing, The Hang Seng University of Hong Kong. Before that, she had been conducting applied research in Hong Kong Applied Science and Technology Research Institute (ASTRI) as a Senior Lead Engineer on 3G, 4G and 5G mobile communication systems. She also had working experience in Shenzhen University, IBM and Alcatel-Lucent with research focuses on V2X, smart city, 5G, network optimization and machine learning. Her research interests include vehicular communication networks, optimization and machine learning.

Zijian Li received the bachelor's degree from The South China University of Technology, and the M.Sc. degree from The Hong Kong Polytechnic University. Now he is studying for a Ph.D. degree at The Hong Kong University of Science and Technology. He is interested in federated learning and distributed learning. To be specific, his research focuses mainly on data heterogeneity, domain generalization, and privacy protection in federated learning.