FiDo: Ubiquitous Fine-Grained WiFi-based Localization for Unlabelled Users via Domain Adaptation

Xi Chen*
alex.chen1@samsung.com
Samsung AI Center
Montreal, Canada

Hang Li*
hang2.li@partner.samsung.com
Samsung AI Center
Montreal, Canada

Chenyi Zhou
chenyi.zhou@partner.samsung.com
Samsung AI Center
Montreal, Canada

Xue Liu
steve.liu@samsung.com
Samsung AI Center
Montreal, Canada

Di Wu
di.wu1@samsung.com
Samsung AI Center
Montreal, Canada

Gregory Dudek
greg.dudek@samsung.com
Samsung AI Center
Montreal, Canada

ABSTRACT

To fully support the emerging location-aware applications, location information with meter-level resolution (or even higher) is required anytime and anywhere. Unfortunately, most of the current location sources (e.g., GPS and check-in data) either are unavailable indoor or provide only house-level resolutions. To fill the gap, this paper utilizes the ubiquitous WiFi signals to establish a (sub)meter-level localization system, which employs WiFi propagation characteristics as location fingerprints. However, an unsolved issue of these WiFi fingerprints lies in their inconsistency across different users. In other words, WiFi fingerprints collected from one user may not be used to localize another user. To address this issue, we propose a Wi**Fi**-based **Do**main-adaptive system **FiDo**, which is able to localize many different users with labelled data from only one or two example users. FiDo contains two modules: 1) a data augmenter that introduces data diversity using a Variational Autoencoder (VAE); and 2) a domain-adaptive classifier that adjusts itself to newly collected unlabelled data using a joint classification-reconstruction structure. Compared to the state of the art, FiDo increases average F1 score by 11.8% and improves the worst-case accuracy by 20.2%.

CCS CONCEPTS

• **Human-centered computing** → **Ubiquitous and mobile computing systems and tools**; • **Computing methodologies** → **Machine learning**.

KEYWORDS

WiFi-based localization, domain adaptation, data augmentation

ACM Reference Format:
Xi Chen, Hang Li, Chenyi Zhou, Xue Liu, Di Wu, and Gregory Dudek. 2020. FiDo: Ubiquitous Fine-Grained WiFi-based Localization for Unlabelled Users via Domain Adaptation. In *Proceedings of The Web Conference 2020 (WWW '20), April 20–24, 2020, Taipei, Taiwan.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3366423.3380091

*The first two authors contributed equally to this paper.

WWW '20, April 20–24, 2020, Taipei, Taiwan

ACM ISBN 978-1-4503-7023-3/20/04.
https://doi.org/10.1145/3366423.3380091

1 INTRODUCTION

Location Based Services (LBS) not only provide individuals with the information about their surroundings, but also assist businesses to push location-aware updates to their users. By doing these, a wide range of mobile and Internet of Things (IoT) applications are enabled, including geo-social networking, Point-of-Interest (POI) recommendation, local business/product search, smart home automation and wireless security fence. Recently, more advanced location-aware applications, such as cashier-less shopping, targeted ads display, geo-based Augmented Reality (AR), indoor navigation/tracking, and smart home automation, have been rapidly emerging and substantially changing people's lives. Together with their great potentials, these applications also bring stringent requirements to the location information. To fully support them, location information must have at least meter-level resolution, and be available at anytime and anywhere. Unfortunately, the primarily location sources, e.g., check-in data, locations embedded in tweets, and Global Positioning System (GPS) data, either have a low resolution (house-level or even lower) or are unavailable indoor.

To fill this gap, the omnipresent WiFi signals have been utilized to provide ubiquitous location information for everyone with (sub)meter-level resolution. Existing WiFi-based localization systems can be divided into two categories: device-oriented and device-free. The device-oriented systems use pre-measured locations of WiFi Access Points (APs) as references, and triangulate a device carried by the user with dedicated WiFi signals. On the contrary, the device-free systems pinpoint the physical presence of a user directly, by passively learning the changes in WiFi propagation caused by a human body. These systems require no dedicated equipment nor extra WiFi overhead. In this sense, device-free systems are more flexible, light-weight and easy-to-deploy. Concretely, they record WiFi propagation characteristics, such as Received Signal Strength Indicator (RSSI) [47] and Channel State Information (CSI) [5] when users are at different locations. These propagation characteristics are then associated with location labels/coordinates to create device-free WiFi location fingerprints. Later, the systems match real-time propagation characteristics with those recorded in the fingerprints, so as to tell the locations of users on-the-fly.

However, WiFi location fingerprints may experience significant inconsistency across different users. A localization system trained on one user's fingerprints may very likely not work for another one with a different body shape. (More details are to be discussed in Section 2.3.) To solve this inconsistency issue and provide robust location information for different users, this paper proposes a Wi**Fi**-based **Do**main-adaptive localization system **FiDo**, which is able to localize new users without labelling their data and thus saves a huge amount of system calibration time and efforts. This task is difficult, since we need to address two practical challenges.

Challenge 1: It is unrealistic to create a comprehensive fingerprint database that covers all possible users.

To address Challenge 1, FiDo utilizes a **data augmenter** to generate synthetic fingerprints that are similar to yet different from the collected ones. This augmenter uses Variational Autoencoders (VAEs) to summarize the statistical features of the WiFi fingerprints. It then generates synthetic fingerprints, which are samples from the random distribution described by the statistical features. By doing the above steps, FiDo augments the fingerprints collected from a small group of example users (e.g., one or two) to a large group of virtual users, and thus is able to cover more users.

Challenge 2: After the system being deployed, labelling new data (i.e., associating newly recorded WiFi characteristics with locations) is troublesome and impossible most of the time.

Concretely, to label new users, it is required to associate CSI readings with ground-truth locations. This ground-truth is unavailable after system deployment, unless other localization systems, e.g., LiDARs, are installed and synchronized with the WiFi-based system.

To tackle Challenge 2 and utilize the potentially large amount of unlabelled data, FiDo learns a **domain-adaptive classifier**, which is based on a special Neural Network (NN) with a joint classification-reconstruction structure. A classification path first extracts features from labelled data and then predicts a location based on these features. A reconstruction path implements a reconstruction Autoencoder (AE), of which the output tries to reproduce the input. This AE aims to learn a general representation of both labelled and unlabelled data. These two paths are integrated in a way that they share the same feature extraction layers. In this way, after the unlabelled data tune the parameters of feature extraction layers, the classification path is forced to predict locations with features representing unlabelled domains. As a result, FiDo learns to predict locations in different domains covering both WiFi fingerprints (i.e., labelled data) from example users and newly collected WiFi data (i.e., unlabelled data) from previously unseen users.

It is worth noticing that FiDo is different from domain-adversarial NNs. The existing domain-adversarial NNs have an assumption, which is: the NNs know whether a data point comes from the target domain or the source domain. In other words, for these NNs, the so-called unlabeled data still have implicit labels to tell the domains. Yet, in the practice of localizing different people, we usually cannot have such a domain label (i.e., a user ID), as it requires every user to identify herself/himself to the system. In contrast to domain-adversarial NNs, FiDo does not require such a domain label. All unlabelled data points will go through the reconstruction part with no indication of domains.

To evaluate FiDo, we established a (sub)meter-level localization testbed using Commercial Off-The-Shelf (COTS) WiFi devices. We invited 9 volunteers with different heights, weights, and genders, to participate in our experiments. We recorded WiFi CSI data when they were standing at different locations. The minimum distance between two locations is $70cm$. Compared to the state of the art, FiDo was able to improve the average recall, precision and F1 score by 11.7%, 9.2% and 11.8%, respectively. It also boosted the worst-case accuracy across different people by 20.2%, indicating that FiDo is much more robust than the state of the art in the presence of different body shapes. Moreover, FiDo was able to improve its own performance, as more and more unlabelled data were collected.

2 BACKGROUND AND MOTIVATIONS

2.1 Advances in Location-Aware Applications

Location-aware applications have recently attracted extensive interest from many online businesses, especially from those associated with the Location-Based Social Networks (LBSN). Location data can be collected from a vast variety of sources, including the traditional GPS, user check-ins, or even casual mentions in text-based tweets [9, 45]. These data can be efficiently used for recommendation systems in friendship proposing and trip planning [8]. More recently, the development of new applications brings the need of fine-grained and ubiquitous location information. For example, cashier-less shopping projects need to locate customers with respect to individual shelves [2]. In shopping malls, an accurate navigation system would be helpful to direct customers through the multi-floor space. In addition, the Augmented and Virtual Realities (AR/VR) often require users' locations on a meter-level or even sub-meter-level, so as to render virtual images precisely [11, 34]. The aforementioned location sources are not enough to support them.

2.2 WiFi-based Indoor Localization

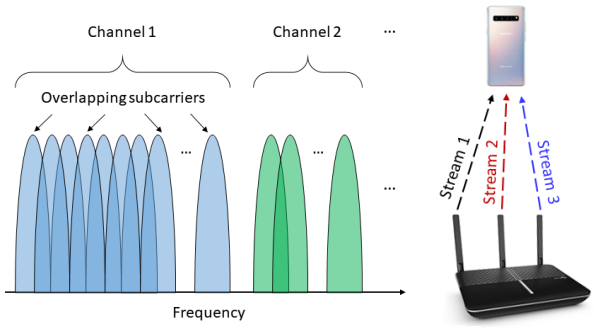
In order to provider meter-level localization services anytime and anywhere, the ubiquitous WiFi signals have been utilized.

The first kind of WiFi-based approaches aim to triangulate a WiFi device using the Time of Flight (ToF) [36, 42] and/or the Angle of Arrival (AoA) [19] from the device to multiple APs. These device-dependent approaches heavily rely on a priori measurements of AP locations, as well as the orientations of on-hand devices.

The second and more promising approach is to localize a human's body directly, instead of a device. Such a device-free approach [5] leverages a physical phenomenon: a human body introduces different WiFi propagation characteristics at different locations, by absorbing some signals while bouncing back others. Running on a plug-in WiFi equipment (e.g., an AP or a smart TV), a device-free approach passively listens to the WiFi traffic already in the air, extracts signal propagation characteristics such as RSSI and CSI, and associates these characteristics with location labels to create fingerprints. Later, by matching the real-time WiFi characteristics with the collected fingerprints, we are able to localize a user, even if he/she carries no device. These approaches work anywhere indoor and outdoor, as long as there is a WiFi coverage.

2.3 WiFi Channel State Information

Among all WiFi propagation characteristics, CSI is the most informative one. According to WiFi physical layer standard [14], a WiFi



(a) WiFi frequency subcarriers. (b) WiFi spatial streams.

Figure 1: WiFi frequency and spatial multiplexing.

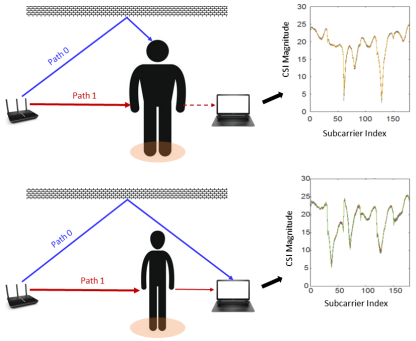


Figure 2: Different WiFi CSI magnitudes caused by different people standing at the same location.

frequency channel is divided into multiple (64, 128 or even more) frequency subcarriers using the Orthogonal Frequency-Division Multiplexing (OFDM) technology, as presented in Fig. 1(a). In addition, WiFi also supports spatial multiplexing, i.e., transmitting separate data signals from and/or to multiple antennas. The signal flow from one transmitting antenna to one receiving antenna is called a spatial stream, or stream for short. One example of spatial multiplexing is illustrated in Fig. 1(b).

Here we use $a_{i,j}$ to denote the signal transmitted on the ith stream and the jth subcarrier. At the other side, the receiver obtains a signal $b_{i,j}$, which is the transmitted signal $a_{i,j}$ altered by the wireless propagation. Such an alteration can be represented by a complex-valued channel response $h_{i,j}$. Therefore, in the frequency domain, the alteration is expressed as $b_{i,j} = h_{i,j} \cdot a_{i,j}$. For the receiver to reconstruct the original data $a_{i,j}$, it needs to cancel out $h_{i,j}$ from the received signal $b_{i,j}$.

A major drawback of WiFi-based location fingerprints lies in their inconsistency across different people. For example, as shown in Fig. 2, a tall person blocks WiFi propagation path 0, while a short person doesn't. Meanwhile, a big person absorbs more signal power on path 1, while a small person absorbs less. As a result, the WiFi propagation characteristics (in this example we use CSI magnitudes) vary largely across different people, even when they stand at the same location. Trained by one user's data, a localization system is very likely to perform terribly for other unseen users.

There are seemingly easy ways to solve this issue. Ideally, if we can collect location fingerprints from all possible users at every location with meter-level resolution, we can train a generic system that is robust and accurate for everyone. Unfortunately, given the diversity of human's bodies, this will cost tremendous time and efforts, and thus is impractical (i.e., **Challenge 1**). Alternately, if we can label WiFi signals every time when a new user shows up, then we can use these new fingerprints to update a deployed localization system. However, this requires a new user to manually associate her WiFi CSI (as well as her identification) with real-time locations, which she may not even know (i.e., **Challenge 2**).

3 DESIGN OVERVIEW

In order to address the above-mentioned challenges, in this section, we propose our FiDo system.

3.1 Definitions

Suppose one WiFi device is sending signals using its N_{TX} transmission antennas. Another WiFi device is receiving these signals using its N_{RX} receiving antennas. In this sense, there are in total $N_{SS} = N_{TX} \times N_{RX}$ spatial streams. Suppose each WiFi frequency channel is divided into N_{SC} frequency subcarriers via OFDM. Let $h_{i,j}[t]$ denote the complex WiFi CSI value of the ith stream on the jth subcarrier collected at time t. Let $\mathbf{h}_i[t]$ denote the complex vector of CSI on the ith stream collected at time t, i.e.,

$$\mathbf{h}_i[t] = \left(h_{i,1}[t], h_{i,2}[t], \cdots, h_{i,N_{SC}}[t]\right). \tag{1}$$

Let $\mathbf{H}[t]$ denote the complex vector of CSI on all the subcarriers and streams collected at time t as

$$\mathbf{H}[t] = \left(\mathbf{h}_1[t], \mathbf{h}_2[t], \cdots, \mathbf{h}_{N_{SS}}[t]\right). \tag{2}$$

In this work we set up one transmission antenna and three receiving antennas, namely $N_{TX} = 1$, $N_{RX} = 3$, and thus $\mathbf{H}[t]$ has 90 elements in total.

Denote the data inputted to the localization system at time t as $\mathbf{x}[t]$. In this paper, we use the magnitudes of CSI as the data, i.e.,

$$\mathbf{x}[t] = \left(|h_{1,1}[t]|, \cdots, |h_{1,N_{SC}}[t]|, \cdots, \left|h_{N_{SS},N_{SC}}[t]\right|\right), \tag{3}$$

where $|\cdot|$ denotes the magnitudes (i.e., absolute values) of complex numbers. We use Ω to represent the finite set of locations supported by the proposed framework. We then use $y[t] \in \Omega$ to label a user's ground-truth location at time t. In addition, we represent y_t using one-hot encoding to achieve a one-hot label vector $\mathbf{y}[t]$, of which the dimension is the number of supported locations N_Ω.

If an input data point (i.e., a CSI magnitude vector) is associated with a location label, then we call it a labelled data point, denoted as $\mathbf{x}_L[t]$. Otherwise, we call it an unlabelled data point, denoted as $\mathbf{x}_U[t]$. Let $X_L = \{\mathbf{x}_L[t]\}$ denote all the labelled data already collected, and let $X_U = \{\mathbf{x}_U[t]\}$ denote all the unlabelled data already collected. A location fingerprint $f[t]$ is then defined a tuple of a labelled data point and its location label, i.e.,

$$f[t] = (\mathbf{x}_L[t]; \mathbf{y}[t]). \tag{4}$$

Let $F = \{f[t]\}$ denote the set of all collected fingerprints. And we use $Y = \{\mathbf{y}[t]\}$ to denote all the ground-truth location labels.

The **mission** of FiDo is to train a location classifier using F and X_U, so that we can predict the $\mathbf{y}[t]$ accurately given a new $\mathbf{x}[t]$.

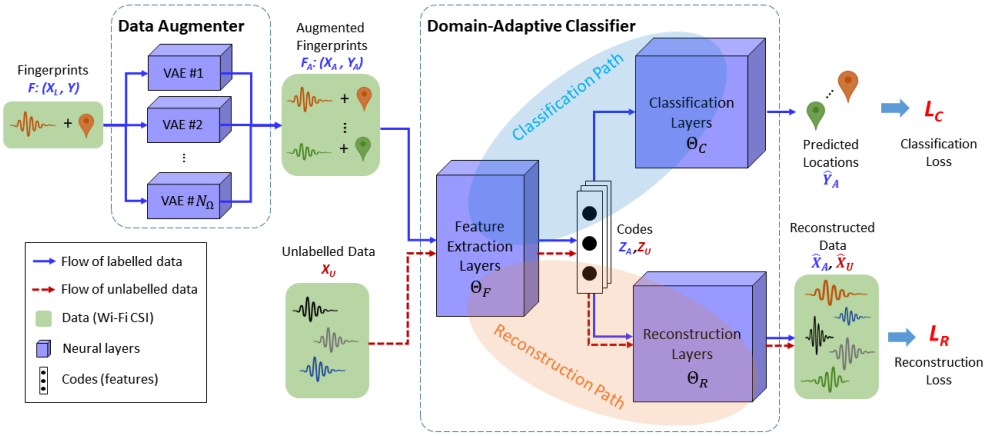


Figure 3: An overview of the FiDo

3.2 Modules of FiDo

As shown in 3, FiDo contains two major modules: the data augmenter and the domain-adaptive classifier.

From the leftmost side of Fig. 3, the fingerprints F first enter the data augmenter, which generates synthetic fingerprints from the existing ones. A synthetic fingerprint $f_S[t]$ is defined as:

$$f_S[t] = (\mathbf{x}_S[t]; \mathbf{y}_S[t]), \tag{5}$$

where $\mathbf{x}_S[t]$ is a synthetic data point from the synthetic data set X_S, and $\mathbf{y}_S[t] \in \Omega$ is a synthetic location labelled associated with $\mathbf{x}_S[t]$. Also, let $Y_S = \{\mathbf{y}_S[t]\}$ denote all synthetic labels, and let $F_S = \{f_S[t]\}$ denote the set of all synthetic fingerprints. The final output F_A is then a disjoint union of F_S and F, i.e.,

$$F_A = F \sqcup F_S. \tag{6}$$

By doing this, we can extend our collected data F to cover more potential users with F_A, and thus address **Challenge 1**. More details of the data augmenter will be discussed in Section 4.

These augmented fingerprints F_A, together with the unlabelled data X_U will be used to train the domain-adaptive classifier. To achieve the domain adaptation capability, this classifier embraces a joint classification-reconstruction structure. As shown in Fig. 3, the classifier consists of two paths: the classification path and the reconstruction path. On one hand, the classification path contains feature extraction layers followed by the classification layers. This path only takes labelled data (i.e., the augmented data X_A) as input. On the other hand, the reconstruction path shares the same feature extraction layers with the classification path, and then diverges into separate reconstruction layers. This path takes both labelled data (i.e., X_A) and unlabelled data (i.e., X_U) as input.

For the classification path, when an augmented fingerprint $f_A[t]$ comes, it passes the data part $\mathbf{x}_A[t]$ through the feature extraction layers to calculate a feature vector $\mathbf{z}_A[t]$. From this feature vector, the classification layers predict a location probability vector $\hat{\mathbf{y}}_A[t]$. Let $\hat{Y}_A = \{\hat{\mathbf{y}}_A[t]\}$ denote the predicted probability vectors for all augmented data points in X_A. The **classification loss** of the domain-adaptive classifier is defined as:

$$L_C = \sum_t l_C(\mathbf{y}_A[t], \hat{\mathbf{y}}_A[t]), \tag{7}$$

where $l_C[t]$ is the cross entropy loss of a single data point as:

$$l_C(\mathbf{y}_A[t], \hat{\mathbf{y}}_A[t]) = -\sum_i \rho_i[t] \log \hat{\rho}_i[t]. \tag{8}$$

$\rho_i[t]$ and $\hat{\rho}_i[t]$ are the ith elements of $\mathbf{y}_A[t]$ and $\hat{\mathbf{y}}_A[t]$, respectively.

For the reconstruction path, both $\mathbf{X}_A$ (with labels) and $\mathbf{X}_U$ (without labels) will flow through it. If an unlabelled data point $\mathbf{x}_U[t]$ comes, the reconstruction path first passes $\mathbf{x}_U[t]$ through the shared feature extraction layers to obtain a feature vector $\mathbf{z}_U[t]$. Given this $\mathbf{z}_U[t]$, the reconstruction layers output a $\hat{\mathbf{x}}_U[t]$, which tries to reproduce $\mathbf{x}_U[t]$ as much as possible. If the incoming data point is a labelled one $\mathbf{x}_A[t]$, then the reconstruction path reuses the feature vector $\mathbf{z}_A[t]$, and generates another $\hat{\mathbf{x}}_A[t]$ to reproduce $\mathbf{x}_A[t]$. We denote reconstructed data as $\hat{X}_A$ and $\hat{X}_U$ for augmented and unlabelled data, respectively. We then define the **reconstruction loss** of the domain-adaptive classifier as:

$$L_R = \sum_t d(\mathbf{x}_A[t], \hat{\mathbf{x}}_A[t]) + \sum_t d(\mathbf{x}_U[t], \hat{\mathbf{x}}_U[t]), \tag{9}$$

where $d(\cdot, \cdot)$ is a distance metric between two vectors. In this paper, $d(\cdot, \cdot)$ is implemented as the Euclidean distance.

By sharing the feature extraction layers between both paths, we force the classification layers to use features that are representative for both labelled and unlabelled data. In this way, the classifier can adapt itself to unlabelled domains, where the unlabelled data come from. Although we cannot conduct data labelling on new domains, we can still leverage the unlabelled data to adapt a classifier to these domains. Therefore, **Challenge 2** is solved. More details of the domain-adaptive classifier is to be presented in Section 5.

3.3 Hyper-parameters

We conduct a coarse-grained search to find out a suitable set of hyper-parameters. We believe that a fine-grained hyper-parameter tuning can indeed provide some (marginal) improvements, and leave it to our future work. Another thing to note is that, we use relatively shallow NNs for both data augmenter and domain-adaptive classifier. The reasons are two-fold. First, considering the labeling effort, it is very unlikely to collect enough data to train a decent deep solution. Secondly, shallow NNs can adapt faster to new domains with a small number of new data points, and thus are able to support applications with stringent timeliness requirements.

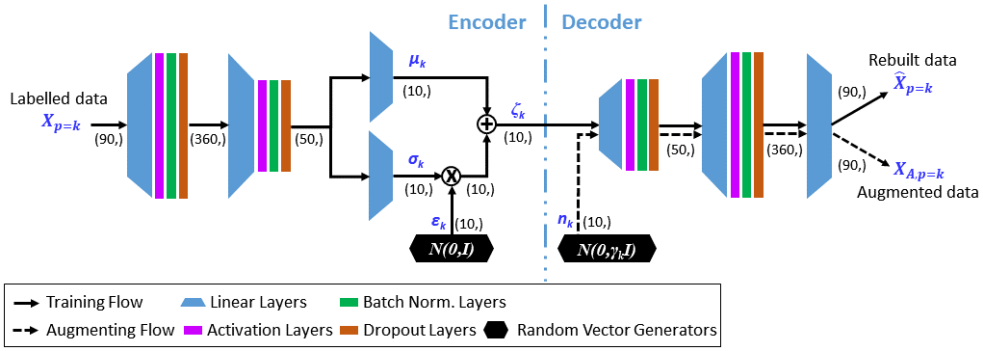


Figure 4: Variational autoencoder for data augmentation.

4 DATA AUGMENTER

To generate more fingerprints from the collected ones, the data augmenter implements N_Ω VAEs. In our design, each VAE is dedicated to one (system-supported) location. The kth VAE is trained with fingerprints collected at the kth location, and produces synthetic fingerprints for this location only.

As illustrated in Fig. 4, a VAE has an encoder-decoder structure, which is similar to its precedents. Yet, different from a traditional AE, the encoder of a VAE aims to encode the input into a normal distribution $N(0, \gamma_k I)$, where γ_k is a pre-selected hyperparameter and I is an identity matrix.. Correspondingly, the decoder tries to reproduce the input from this normal distribution. In this way, we can feed samples from the distribution $N(0, \gamma_k I)$ into the decoder, and generate a large amount of synthetic data similar to yet different from the training data.

4.1 Training a VAE

During the training phase, the data augmenter separates the fingerprints into N_Ω subsets according to their location labels. We use $F_{p=k}$ to denote the fingerprint subset of the kth ($k = 1, 2, \cdots, N_\Omega$) location, and denote the data part of these fingerprints as $X_{p=k}$. $X_{p=k}$ is then fed to the kth VAE for training. The kth VAE takes a batch $X^b_{p=k}$ from $X_{p=k}$, and feeds this batch into its encoder. For the sake of clarity, we describe the data flow using one single data point $\mathbf{x}_k[t] \in X^b_{p=k}$ as an example.

From the 90-dimensional input $\mathbf{x}_k[t]$, the encodes extracts a 10-dimensional vector of means $\mu_k[t]$ and a 10-dimensional vector of standard deviations $\sigma_k[t]$. These two vectors define a 10-dimensional normal distribution, from which a latent vector $\zeta_k[t]$ is drawn. In order to preserve the back propagation machinery, we apply the reparameterization trick [17] to generate $\zeta_k[t]$:

$$\zeta_k[t] = \mu_k[t] + \varepsilon_k[t] * \sigma_k[t], \tag{10}$$

where $\varepsilon_k[t]$ is drawn from the normal distribution $N(0, I)$, with I being 10-dimensional. Eq. (10) is implemented by inserting two elementwise arithmetic nodes in the network model, as demonstrated in Fig.4. Recall that the task of the encoder is to match the distribution of $\zeta_k[t]$ with the normal distribution $N(0, \gamma_k I)$. Hence, we define the loss between the above distributions as the Kullback–Leibler divergence

$$l_{\text{KLD}}[t] = \mathcal{D}\left(N(\mu_k[t], \sigma_k[t]), N(0, \gamma_k I)\right). \tag{11}$$

The output of the encoder (i.e., $\zeta_k[t]$) is then passed to the decoder, which generates a vector $\hat{\mathbf{x}}_k[t]$ to rebuild the input $\mathbf{x}_k[t]$. The rebuilding loss[1] is defined as the Euclidean distance between $\mathbf{x}_k[t]$ and $\hat{\mathbf{x}}_k[t]$:

$$l_{\text{B}}[t] = d(\mathbf{x}_k[t], \hat{\mathbf{x}}_k[t]). \tag{12}$$

The loss function of the whole VAE model is defined as the sum of the distribution loss and the rebuilding loss [17], i.e.,

$$l_{\text{VAE}}[t] = l_{\text{KLD}}[t] + l_{\text{B}}[t]. \tag{13}$$

4.2 Generating Synthetic Data

The generation of synthetic data is conducted solely by the decoder. It starts from a sampler, which generates a 10-dimensional sample $\mathbf{n}_k[t]$ from a normal distribution $N(0, \gamma_k I)$ in every iteration. This $\mathbf{n}_k[t]$ is then passed through the decoding layers, which produce a 90-dimensional vector $\mathbf{x}_{S,p=k}[t]$. We keep producing new synthetic data points, until the size of the synthetic data for the kth location $X_{S,p=k} = \{\mathbf{x}_{S,p=k}[t]\}$ is 10 times larger than the size of the original labelled data $\mathbf{X}_{p=k}$.

4.3 Auto-Labelling

The 1-on-1 VAE-location mapping allows each VAE to automatically label the synthetic data points using the location dedicated to this VAE. Concretely, once the kth VAE produces a synthetic data point $\mathbf{x}_{S,p=k}$, a location vector $\mathbf{k}$ (the one-hot encoding of k) is assigned to it to create a synthetic fingerprint as

$$f_S[t] = \left(\mathbf{x}_{S,p=k}[t]; \mathbf{k}\right). \tag{14}$$

We collect all synthetic data points, label them based on Eq. (14), and obtain the synthetic fingerprints as $F_S = \{f_S[t]\}$. The final output F_A is then the disjoint union of F_S and F, as in Eq. (6).

5 DOMAIN-ADAPTIVE CLASSIFIER

In this section, we discuss the detailed design of the domain-adaptive classifier, and present a new algorithm to train it.

5.1 Feature Extraction Layers

The feature extraction layers are the starting point of both the classification path and the reconstruction path. All augmented (labelled) and unlabelled data will pass through these layers to

[1]This rebuild loss is different from the reconstruction loss of the domain-adaptive classifier to be presented later in Section 5.

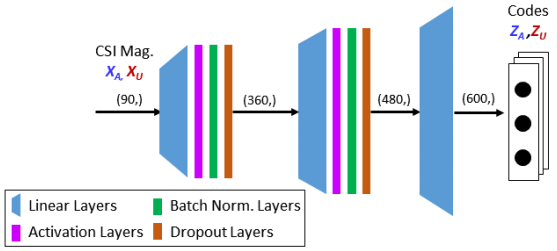


Figure 5: Feature Extraction Layers.

generate their feature vectors. Next we take augmented data $\mathbf{X}_A$ as an example to describe how data flows through these layers. The flow of unlabelled data $\mathbf{X}_U$ is the same in these layers.

Fig. 5 illustrates this data flow, from left to right. An augmented data point $\mathbf{x}_A[t] \in \mathbf{X}_A$ enters the first linear layer, which extends this 90-dimensional vector into a 360-dimensional vector. The second linear layer extends the 360-dimensional vector into a 480-dimensional vector. The last linear layer further extends the dimensions from 480 to 600, and outputs the feature vector $\mathbf{z}_A[t]$. The above three layers compute linear combinations of CSI magnitudes on different subcarriers and streams to generate meaningful features. In addition, we insert a sigmoid activation layer, a batch normalization layer and a dropout layer (with a dropout rate of 0.3) between the first and the second linear layers, as well as between the second and the last linear layers. The sigmoid layers aim to introduce some non-linearity; the batch normalization layers speed up the training; and the dropout layers try to avoid overfitting.

5.2 Classification Layers

After the feature vectors are extracted, the classification path flows into the classification layers. During the training phase, these layers are only opened to feature vectors extracted from augmented data, i.e., $\mathbf{Z}_A$. Fig. 6 presents how a feature vector $\mathbf{z}_A[t]$ passes through the classification layers to generate a location prediction.

Concretely, a 600-dimensional feature vector $\mathbf{z}_A[t]$ is shrunk to a 300-dimensional vector and then a 100-dimensional vector, by the first and the second linear layers, respectively. This 100-dimensional vector is then passed to the third linear layer to predict a N_Ω-dimensional one-hot encoded location probability vector $\hat{\mathbf{y}}_A[t]$. Given the vector $\hat{\mathbf{y}}_A[t]$, we calculate the classification loss $l_C[t]$ of one data point according to Eq. (8). Then, the aggregated classification loss L_C of all augmented data (or a batch of augmented data) is computed according to Eq. (7).

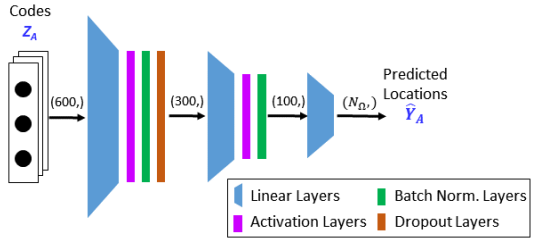


Figure 6: Classification Layers.

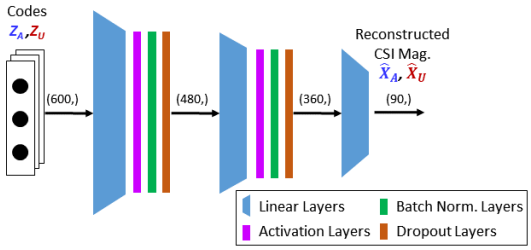


Figure 7: Reconstruction Layers.

5.3 Reconstruction Layers

After coming out from the feature extraction layers, the reconstruction path flows into the reconstruction layers. These layers only operate during training, and take both augmented and unlabelled data as inputs. Next we describe the data flow of unlabelled data as an example. The flow of the augmented data is identical.

As shown in Fig. 7, a 600-dimensional feature vector $\mathbf{z}_U[t]$ is shrunk to a 480-dimensional vector and later a 360-dimensional vector by the first and the second linear layers, respectively. From this 360-dimensional vector, the last linear layer tries to generate a 90-dimensional vector $\hat{\mathbf{x}}_U[t]$ that reproduces $\mathbf{x}_U[t]$ as much as possible. An aggregated reconstruction loss is computed for all data with and/or without labels (or a batch of unlabelled and/or augmented data points) using Eq. (9).

5.4 Alternating Training

The aforementioned layers form a joint classification-reconstruction structure, with the shared feature extraction layers as the joint of the two paths. Under this structure, the extracted features are representative for both labelled and unlabelled data, and assist the classifier to extend its coverage to the unlabelled domains (i.e., unlabelled users in this paper). In this way, even when the newly collected

Algorithm 1: Alternating Training

Input : Augmented data X_A with labels Y_A, unlabelled data X_U, step sizes α_1 and α_2.
Output : Trained parameters Θ_F, Θ_C and Θ_R.
while *not done* **do**
  Sample a batch X_A^b from X_A, and a batch X_U^b from X_U;
  Take out X_A^b's corresponding labels Y_A^b from Y_A;
  Training the Classification Path:
    Pass X_A^b through Θ_F and Θ_C to obtain $\hat{Y}_A^b$;
    Evaluate classification loss $L_C(Y_A^b, \hat{Y}_A^b)$ by Eq. (7);
    $(\Theta_F, \Theta_C) \leftarrow (\Theta_F, \Theta_C) - \alpha_1 \frac{dL_C}{d(\Theta_F, \Theta_C)}$;
  end
  Training the Reconstruction Path:
    Pass X_A^b through Θ_F and Θ_R to obtain $\hat{X}_A^b$;
    Pass X_U^b through Θ_F and Θ_R to obtain $\hat{X}_U^b$;
    Evaluate reconstruction loss $L_R(X_A, \hat{X}_A, X_U, \hat{X}_U)$ by Eq. (9);
    $(\Theta_F, \Theta_R) \leftarrow (\Theta_F, \Theta_R) - \alpha_2 \frac{dL_R}{d(\Theta_F, \Theta_R)}$.
  end
end

data cannot be labelled, FiDo still automatically learns to utilize the features of new users. In order to train this joint structure, we develop an alternating training algorithm. For clarity, we denote the parameters of the feature extraction layers, the classification layers and the reconstruction layers as Θ_F, Θ_C and Θ_R, respectively.

The alternating algorithm is summarized as Algorithm 1. Given the augmented fingerprints F_A (i.e., $[X_A; Y_A]$) and the unlabelled data X_U, we first sample a batch of labelled data points X_A^b from X_A (and the corresponding Y_A^b from Y_A), and a batch of unlabelled data point X_U^b from X_U. We first train the classification path with X_A^b to update both Θ_F and Θ_C, and then train the reconstruction path with both X_A^b and X_U^b to update Θ_F and Θ_R. We then iterate the training processes of two paths. As a result, Θ_F is trained to capture features that are able to 1) represent data from both labelled and unlabelled domains and 2) classify locations precisely.

6 EVALUATION

6.1 Experiment Setup

6.1.1 Environment and Devices. The experiment environment and device placements are illustrated in Fig. 8. We conducted the experiments in a $3m \times 4m$ office room. All tested localization systems were expected to cover 8 locations (i.e., $N_\Omega = 8$), which were denoted as $p0, p1, \cdots, p7$, respectively. Among them, $p0$ is a virtual location corresponding to a status that the room is empty [5]. The rest locations were distributed inside the room with a minimum distance of $70cm$. A Dell Latitude E7440 laptop with Intel 5300 WiFi chip was connected to a TP-Link AC1750 WiFi router to create standard WiFi signals covering the whole room. We installed the Linux CSI Tool [12] on the laptop to extract CSI readings and feed them to FiDo (and other systems), which also resides on the laptop. To collect CSI readings, we turned on 1 antenna (out of 3) on the WiFi router, and enabled all 3 antennas on the laptop. The Intel 5300 WiFi chip measures CSI on 30 subcarriers. Hence, in the experiments, we had $N_{TX} = 1$, $N_{RX} = 3$, $N_{SS} = 1 \times 3 = 3$, and $N_{SC} = 30$. The CPU (Intel-i5 1.9GHz) utilization was less than 15% and 10% for training and testing, respectively. Memory usage was less than $60MB$. To store a trained model, it took around $4.9MB$ disk space.

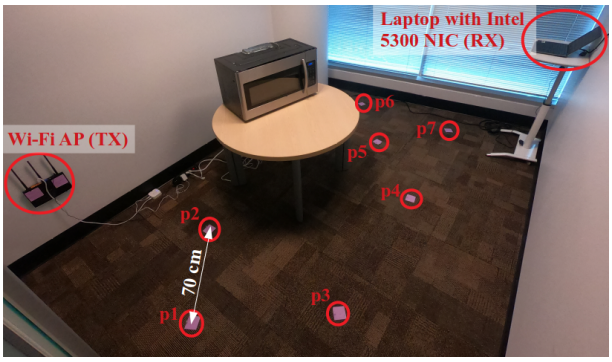


Figure 8: Experiment environment and device placements. $p0$ was a virtual location representing the empty room status, and $p1, \cdots, p7$ were real locations where volunteers stood.

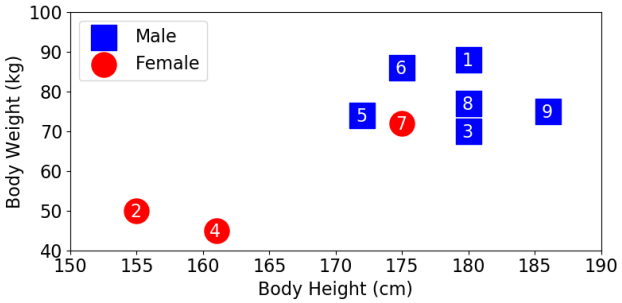


Figure 9: Height and weight distributions of the volunteers.

6.1.2 Diversity of People. To record CSI of different users, we recruited 9 volunteers with different genders (3 females and 6 males), heights (from $155cm$ to $186cm$) and weights (from $45kg$ to $88kg$). Their distributions are illustrated in Fig. 9.

6.1.3 Data Collection. We instructed the router to periodically send pings to the laptop, with a speed of 100 packets per second. This corresponds to a 51.2 *Kbps* traffic, consuming only a tiny fraction of WiFi capacity. We asked each volunteer to stand at every location for 60 seconds, recorded the WiFi CSI readings, and labelled these readings by this location. During the recording, a volunteer was allowed to move freely and naturally (e.g., turning around, stretching, doing arms akimbo, etc.), as long as a he/she remained on the location. At every location, we repeated this recording process 5 runs for every volunteer. This gave us roughly $100 \times 60 \times 5 = 30K$ CSI readings for one volunteer at one location. Thus, we had roughly $30K \times 8 = 240K$ CSI readings for each volunteer, and $240K \times 9 = 2.16M$ CSI readings in total.

6.1.4 Data Separation. To emulate real-world challenges faced by localization systems (i.e., Challenge 1 and Challenge 2), we separated the data as follows. First, the labelled data used for training only comes from two volunteers, who are called example users. These two example users together provide a limited number of labelled data points, which is $5K$ labelled CSI readings in our experiments. Secondly, to emulate the potentially huge number of unlabelled data, we separate the rest of the data into 50% unlabelled training data (by not using their labels in training) and 50% testing data. Table 1 summarizes the above data separation. The above data separation is summarized in Table 1.

	User(s)	Labelled data	Unlabelled data
Training	ID2&ID4	$5K$	$\sim 1.08M$
Testing	ID1-ID9	$\sim 1.08M$	n/a

Table 1: Data separation.

6.2 Systems Evaluated

To avoid comparing apples to oranges, we only evaluated the following device-free fingerprint-based localization systems that require no domain labels (i.e., user IDs are not available).

- **AutoFi** [5] was used as the baseline[2]. It adapted itself to a new domain based on an empty-room status detection. Once the room was detected as empty using CSI variances, AutoFi updated a

[2]Other existing solutions either require labelled data from new users/domains (e.g., EI [16]) or need pre-measurements of AP locations (e.g., WiDAR [26, 27]). To avoid comparing apples to oranges, we take AutoFi [5] as the baseline.

mapping function from the old domain to the new one, and used this function to adjust its classifier. AutoFi was designed to work with labelled data only.

• **VAE-only** implemented only the VAEs of FiDo, and connected them directly to AutoFi's classifier. Hence, same as AutoFi, VAE-only could only leverage labelled data.

• **DAC-only** implemented only the domain-adaptive classifier, and was able to utilize both labelled and unlabelled training data.

• **FiDo** was the proposed system, which utilized both labelled and unlabelled training data.

6.3 Accuracy

We first took two smallest volunteers (i.e., ID2 and ID4) as example users (of whom $5K$ data points were used as labelled training data). We further separated the data as shown in Table 1. FiDo and DAV-only were able to leverage both the $5K$ labelled training data and the $1.08M$ unlabelled training data. Note that AutoFi and VAE-only could only use the $5K$ labelled data to train themselves.

Table 2 lists the average recall, precision, and F1 score (i.e., the harmonic mean of precision and recall) across all locations and IDs. It is shown that AutoFi had the worst performance, due to its incapability to handle the diversity of users. AutoFi yielded recall and precision as low as 72.9% and 75.8%, respectively. By augmenting the limited labelled data, VAE-only was able to cover more potential users, and thus increased recall and precision to 81.0% and 81.1%, respectively. Meanwhile, DAC-only leveraged the unlabelled data to adapt itself to unlabelled users, and consequently improved recall and precision to 80.8% and 81.0%, respectively. FiDo embraced both the augmentation of labelled data and the domain adaptation with unlabelled data, and thus achieved the best performance among all. Compared to the state-of-the-art AutoFi, FiDo improved recall, precision and F1 scores (in absolute numbers) by 11.7%, 9.2% and 0.118, respectively.

To better compare the localization systems, Fig. 10 presents the location confusion matrices of different systems. It is illustrated that AutoFi's True Positive Rates (TPR) across different locations were quite diverged, ranging from 52.3% at location $p3$ to 95.6% at location $p2$. Such a large gap between the best and the worst (i.e., 43.3%) was an indicator that AutoFi was overfitted by the data from example users. The data augmenter helped VAE-only increase the worst-case TPR to 62.2% at $p1$, while keeping the best TPR as 95.6% at $p2$. However, the 33.4% best-worst TPR gap suggested that VAE-only might be again overfitted. On the contrary, the domain-adaptive classifier design assisted DAC-only to tackle the overfitting, and reduced the best-worst TPR gap to 21.0%. Yet, DAC-only sacrificed the best-case performance with an absolute decrease of 8.4%. By leveraging both data augmenter and domain-adaptive classifier, FiDo was able to claim the benefits of both. Concretely, FiDo improved the worst-case TPR to 80.4%, while maintaining the best-case TPR as high as 93.3%. This means that FiDo not only provided the maximum worst-case performance, but also functioned consistently well across different locations.

	Recall	Precision	F1 score
AutoFi	72.9%	75.8%	0.727
VAE-only	81.0% (+8.1%)	81.1% (+5.3%)	0.807 (+0.080)
DAC-only	80.8% (+7.9%)	81.0% (+5.2%)	0.807 (+0.080)
FiDo	84.6% (+**11.7**%)	**85.0**% (+**9.2**%)	**0.845** (+**0.118**)

Table 2: Average recall, precision and F1 scores of different systems across all locations and IDs.

6.4 Robustness

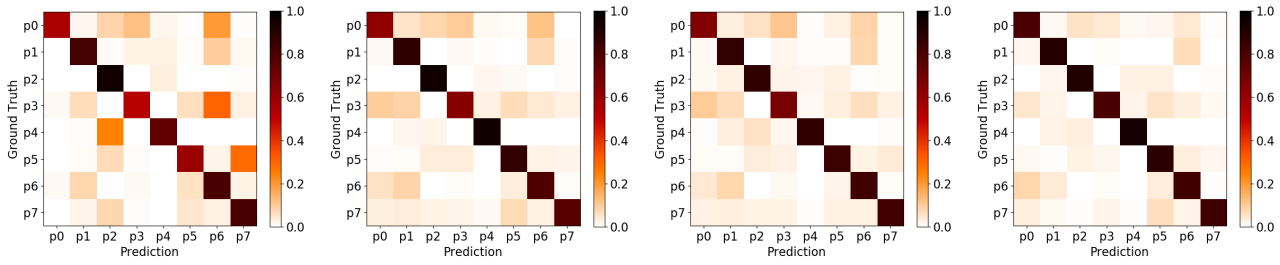
Next we evaluate the robustness of different systems.

6.4.1 Robustness to Users' Heights: We used the average height of example users (i.e., ID2 and ID4) as the example height, and divided all users into three groups based on their absolute differences to this example height. The three groups were $< 5cm$, $5cm \sim 20cm$, and $> 20cm$, respectively. Fig. 11(a) illustrated the localization accuracy values on these three groups. It is confirmed that the localization accuracy decreased with the increase in the height difference between the testing user and the example users. AutoFi was not able to handle this variation in users' heights. When applied to the example users, AutoFi's accuracy is above 95%. However, this number dropped below 67%, if a testing user was $10cm$ taller than the example users. This is the evidence that AutoFi was overfitted by the data of example users. FiDo, VAE-only and DAC-only were able to learn localization features that are less correlated with users' heights, and thus achieved better and more robust performance than the state of the art. FiDo maintained its accuracy above 81% even the height difference is bigger than $20cm$. Therefore, we summarize the following observation:

Observation 1: While existing localization systems (e.g., AutoFi) tend to be overfitted by labelled data, FiDo is able to minimize this effect by updating its classifier with unlabelled data.

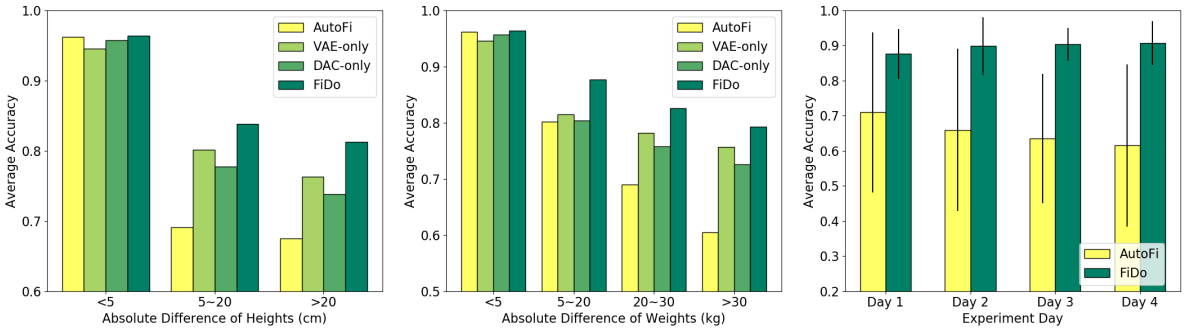
6.4.2 Robustness to Users' Weights: We took the average weight of example users as the example weight, and divided all users into four groups according to their absolute differences to this example weight. The four groups were $< 5kg$, $5kg \sim 20kg$, $20kg \sim 30kg$, and $> 30kg$, respectively. Fig. 11(b) illustrated that, in general, localization accuracy decreased, when the absolute weight difference between example users and testing users increased. Again, we observed that AutoFi was overfitted to the example users. Its accuracy dropped as low as 59.8%, if testing users were $30kg$ heavier than the example users. On the contrary, FiDo had the most robust performance in the presence of diverse user weights. FiDo was able to maintain its accuracy above 80.0%, even when it was trained on the lightest users (i.e., ID2 and ID4) and tested on the heaviest users (i.e., ID1 and ID6). The improvement is 20.2% compared with the state-of-the-art AutoFi.

6.4.3 Robustness to Different Days: To see whether a system's performance is consistent across different days, we asked volunteer ID1 (the most heavy person) to repeat data recording on four different days, i.e., Day 1, 2, 3 and 4. Again, ID2 and ID4 were considered as example users, of whom the data were collected only on Day 1. On Day 1, we trained FiDo and other systems as before, using data separated as in Table 1. From then on, we updated FiDo with ID1's new data every day. Concretely, we saved FiDo's parameters by the end of Day N (where $N = 1, 2, 3$). On Day $N + 1$, we used these saved parameters as the initialization, and learned a new FiDo



(a) AutoFi. (b) VAE-only. (c) DAC-only. (d) FiDo.

Figure 10: Normalized confusion matrices of different systems.



(a) Robustness to users' heights. (b) Robustness to users' weights. (c) Robustness across days (tested on ID1 as an example).

Figure 11: Evaluation of system robustness.

with Algorithm 1 using new unlabelled data of ID1 collected on Day $N+1$ and labelled data of ID2 and ID4 collected on Day 1. It is shown in Fig. 11(c) that localization accuracy of AutoFi decreased with time went by (from 71.0% on Day 1 to 61.5% on Day 4). This decrease was caused by the daily changes (e.g., moved chairs and devices) in the experiment room and the adjacent rooms. To handle these changes, FiDo learned from the daily increasing unlabelled data set, and incrementally adjusted itself in an online manner. As more and more data being collected and learned, FiDo extended its coverage to previously unseen changes, and thus gradually raised its accuracy from 87.6% on Day 1 to 90.7% on Day 4. Therefore, we can summarize the following observation:

Observation 2: FiDo gradually improves its performance by learning from the newly collected (unlabelled) data everyday.

6.5 Convergence of Alternating Training

To demonstrate the convergence of Algorithm 1, we plotted the classification loss L_C and the reconstruction loss L_R during the training phase in Fig. 12. Note that we normalized both losses for better presentation. We can see that, during the first 50 epochs, L_R decreased faster than L_C. At the $50th$ epoch, L_R dropped below 10^{-2}, while L_C still remained above 10^{-1}. Yet, the convergence speed of L_R slowed down from then on. During the period between the $50th$ epoch and the $300th$ epoch, FiDo was trying to learn a good balance between two alternating tasks: classifying the labelled data (i.e., minimizing L_C) and representing the unlabelled data (i.e., minimizing L_R). The parameters of the shared feature layers were alternatively adjusted according to L_C and L_R, and thus both losses fluctuated. At around the $300th$ epoch, FiDo finally stroke

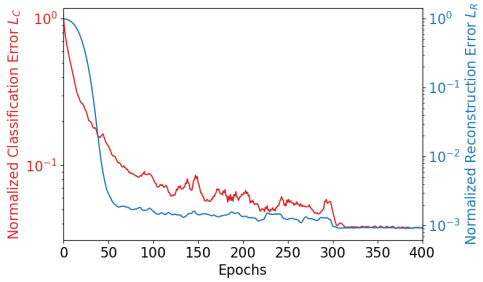


Figure 12: Convergence of Algorithm 1.

a good balance, and hence both L_C and L_R converged quickly, and Algorithm 1 terminated. The above process reflected the close interactions between the classification path and the reconstruction path in FiDo, and produced a strong domain-adaptive classifier.

7 RELATED WORK

Due to the limited space, we only discuss the most relevant work.

7.1 Location-based Services

Location information is the core of numerous online services, such as friendship recommendation in Location-Based Social Networks (LBSN) [46], and trip planning [8, 51]. These services mostly resort to coarse-grained data, e.g., user check-in data [9], locations casually revealed on social media such as twitter [15, 45] and GPS [7]. However, location information from these sources might contain semantic ambiguity and lack the precision for indoor applications.

Fine-grained user localization is required in many thriving indoor applications. For example the Amazon Go project needs to

track both items and customers in its stores [2, 40]. A separate localization system independent of computer vision is also desired for VR and AR [11, 34]. For these applications, a meter-level resolution and ubiquitous availability are critical.

7.2 WiFi-based Localization and Sensing

Classic WiFi localization systems rely on the RSSI, which was used to create location fingerprints for WiFi enabled devices [47]. Although RSSI is relatively easy to obtain and analyze, its localization resolution was usually unsatisfactory.

Instead, geometrical information such as ToF and AoA extracted from CSI largely improved the localization resolution. Based on the ToF of radio signals reflected off human bodies, WiTrack2.0 [1] implemented a multi-person localization system that operates in indoor environments. By combining ToF with AoA measurement, SpotFi [19] identified from all propagation paths the direct ones between the target and the APs, thereby achieving a decimeter-level localization resolution. CUPID2.0 [30] aimed at improving the scalability of its localization system by combining ToF-based trilateration with signal strengths. ToF and AoA based methods have also been implemented in conjunction with the Doppler shift information encoded in the noisy CSI phase factor. As such, the velocity as well as the location of the moving human can be tracked at the same time, achieving a device-free decimeter-level precision [20, 26, 27]. These tracking based methods typically require dedicated devices, tedious pre-measurement of WiFi AP locations, and/or huge efforts in signal preprocessing.

Different from them, the proposed FiDo works with COTS devices and consumes less computing overhead.

More recently, learning methods based on CSI fingerprints have been developed to alleviate users from carrying devices. For example, Sen et al. in [31] developed PinLoc that achieves a 89% meter-level localization accuracy with a clustering-based classification algorithm. Wang et al. in [38] combined the offline training with an online probabilistic data fusion based on the radial basis function. ConFi [4] casted CSI into feature images and feeds them into a Convolutional Neural Network (CNN) that models localization as an image classification problem. Sanam et al. [29] applied Generalized Inter-view and Intra-view Discriminant Correlation Analysis to extract distinguishable CSI features for localization. To tackle the inconsistency of CSI fingerprints due to environmental changes, Chen et al. [5] developed AutoFi to calibrate the localization profiles in an unsupervised manner. Rao et al. in [28] proposed an update scheme to replace the expired fingerprints with new ones.

In this paper, we further push this line of research to the real-life scenarios, and develop the domain-adaptive FiDo that addresses the fingerprint inconsistency issue using only unlabelled data.

Apart from localization, the CSI data have also been utilized for other human centered applications, such as crowd counting [44], gesture recognition [16, 50], and human identification [18, 25]. However, very few efforts have been contributed to a practical solution, which is user- and/or environment-independent.

7.3 Data Augmentation

Data augmentation aims to enhance the diversity and sample size of the training data to improve the final performance of learned models [33]. It is commonly used for image classification tasks [24, 39] and has also shown its effectiveness for several other applications including machine fault detection [32] and recommendation [37].

There are several methods to implement data augmentation. As presented in [33], the general data augmentation methods include translation, rotation, shearing, and flipping of the original data. To further improve the data augmentation speed, auto-augmentation methods have recently been proposed in [6, 21] in which data augmentation is formulated as search problems on data processing functions. Besides the aforementioned methods, generative models have recently been treated as a kind of effective data augmentation method [3, 49]. In [3], the generated images were used to further improve the classification accuracy with an additional metric on whether it is a real image or a generated image. In [49], data augmentation with GANs showed its effectiveness on few-shot learning settings. Different from previous works on using GANs for data augmentation, in our paper, we propose to enlarge the training data set with data generated by VAE models.

7.4 Domain Adaptation and Transfer Learning

Transfer learning [22, 41] aims to improve the learning performance in a data-lacking target domain via transferring some learned knowledge from the data-abundant source domains. Domain adaptation can be treated as a special case of transfer learning that focuses on minimizing the data distribution discrepancy between source and target domains.

Domain adaptation and transfer learning for WiFi localization have been discussed previously. It was shown in [23] and [35] that these tools could help improve the localization accuracy for both time-to-time transfer and device-to-device adaptation. However, their dependency on the labelled data in target domains hinder them from being deployed in practice. FiDo addresses this issue by adapting to target domains with only unlabelled data.

Besides indoor localization, transfer learning has also been applied to many other tasks, including object recognition [10], face recognition [48], electricity load forecasting [43], item ranking [13] and disease forecasting [52].

8 CONCLUSION

In this paper, we develop FiDo, a WiFi based domain-adaptive localization system, which provides ubiquitous (sub)meter-level location information of device-free users. FiDo addresses the inconsistency of WiFi fingerprints across different users, using a data augmenter based on VAEs and a domain-adaptive classifier with a joint classification-reconstruction structure. We propose an alternating training algorithm, which helps FiDo to achieve a perfect balance between classifying the labelled data and extracting domain-independent features from unlabelled data. With the ability to learn from both labelled and unlabelled data, FiDo outperforms the state of the art when applied to different unlabelled users. Moreover, unlike existing domain adversarial solutions, FiDo does not require implicit labels to distinguish target domain data from source domain data, making it more practical to deploy in real-life scenarios. Experiments showed that FiDo not only provided submeter level localization resolution with largely improved accuracy (+20.2% on the worst-case accuracy), but also provided the most robust performance in the presence of diverse unlabelled users.

REFERENCES

[1] Fadel Adib, Zachary Kabelac, and Dina Katabi. 2015. Multi-Person Localization via RF Body Reflections. In *NSDI*. USENIX Association, 279–292.

[2] Amazon. [n.d.]. Amazon Go. www.amazon.com/b?ie=UTF8&node=16008589011.

[3] Antreas Antoniou, Amos Storkey, and Harrison Edwards. 2017. Data augmentation generative adversarial networks. *arXiv preprint arXiv:1711.04340* (2017).

[4] Hao Chen, Yifan Zhang, Wei Li, Xiaofeng Tao, and Ping Zhang. 2017. ConFi: Convolutional Neural Networks Based Indoor Wi-Fi Localization Using Channel State Information. *IEEE Access* 5 (2017), 18066–18074.

[5] Xi Chen, Chen Ma, Michel Allegue, and Xue Liu. 2017. Taming the inconsistency of Wi-Fi fingerprints for device-free passive indoor localization. In *INFOCOM*. IEEE, 1–9.

[6] Ekin D Cubuk, Barret Zoph, Dandelion Mane, Vijay Vasudevan, and Quoc V Le. 2019. AutoAugment: Learning Augmentation Strategies From Data. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 113–123.

[7] Daizong Ding, Mi Zhang, Xudong Pan, Duocai Wu, and Pearl Pu. 2018. Geographical Feature Extraction for Entities in Location-based Social Networks. In *WWW*. ACM, 833–842.

[8] Jie Feng, Yong Li, Chao Zhang, Funing Sun, Fanchao Meng, Ang Guo, and Depeng Jin. 2018. DeepMove: Predicting Human Mobility with Attentional Recurrent Networks. In *WWW*. ACM, 1459–1468.

[9] Jie Feng, Mingyang Zhang, Huandong Wang, Zeyu Yang, Chao Zhang, Yong Li, and Depeng Jin. 2019. DPLink: User Identity Linkage via Deep Neural Network From Heterogeneous Mobility Data. In *WWW*. ACM, 459–469.

[10] Muhammad Ghifary, W Bastiaan Kleijn, Mengjie Zhang, David Balduzzi, and Wen Li. 2016. Deep reconstruction-classification networks for unsupervised domain adaptation. In *European Conference on Computer Vision*. Springer, 597–613.

[11] David Gómez, Paula Tarrío, Juan Li, Ana M. Bernardos, and José R. Casar. 2013. Indoor Augmented Reality Based on Ultrasound Localization Systems. In *PAAMS (Workshops) (Communications in Computer and Information Science)*, Vol. 365. Springer, 202–212.

[12] Daniel Halperin, Wenjun Hu, Anmol Sheth, and David Wetherall. 2011. Tool Release: Gathering 802.11n Traces with Channel State Information. *SIGCOMM Comput. Commun. Rev.* 41, 1 (2011), 53–53.

[13] Guangneng Hu, Yu Zhang, and Qiang Yang. 2019. Transfer Meets Hybrid: A Synthetic Approach for Cross-Domain Collaborative Filtering with Text. In *WWW*. ACM, 2822–2829.

[14] IEEE. [n.d.]. IEEE 802.11n standard. http://standards.ieee.org/findstds/standard/802.11n-2009.html.

[15] Zongcheng Ji, Aixin Sun, Gao Cong, and Jialong Han. 2016. Joint Recognition and Linking of Fine-Grained Locations from Tweets. In *WWW*. ACM, 1271–1281.

[16] Wenjun Jiang, Chenglin Miao, Fenglong Ma, Shuochao Yao, Yaqing Wang, Ye Yuan, Hongfei Xue, Chen Song, Xin Ma, Dimitrios Koutsonikolas, Wenyao Xu, and Lu Su. 2018. Towards Environment Independent Device Free Human Activity Recognition. In *MobiCom*. ACM, 289–304.

[17] Diederik P. Kingma and Max Welling. 2014. Auto-Encoding Variational Bayes. In *2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings*, Yoshua Bengio and Yann LeCun (Eds.). http://arxiv.org/abs/1312.6114

[18] Belal Korany, Chitra R. Karanam, Hong Cai, and Yasamin Mostofi. 2019. XModal-ID: Using WiFi for Through-Wall Person Identification from Candidate Video Footage. In *MobiCom*. ACM.

[19] Manikanta Kotaru, Kiran Raj Joshi, Dinesh Bharadia, and Sachin Katti. 2015. SpotFi: Decimeter Level Localization Using WiFi. In *SIGCOMM*. ACM, 269–282.

[20] Xiang Li, Daqing Zhang, Qin Lv, Jie Xiong, Shengjie Li, Yue Zhang, and Hong Mei. 2017. IndoTrack: Device-Free Indoor Human Tracking with Commodity Wi-Fi. *IMWUT* 1, 3 (2017), 72:1–72:22.

[21] Sungbin Lim, Ildoo Kim, Taesup Kim, Chiheon Kim, and Sungwoong Kim. 2019. Fast autoaugment. *arXiv preprint arXiv:1905.00397* (2019).

[22] Sinno Jialin Pan and Qiang Yang. 2010. A Survey on Transfer Learning. *IEEE Trans. Knowl. Data Eng.* 22, 10 (2010), 1345–1359.

[23] Sinno Jialin Pan, Vincent Wenchen Zheng, Qiang Yang, and Derek Hao Hu. 2008. Transfer learning for wifi-based indoor localization. In *Association for the advancement of artificial intelligence (AAAI) workshop*, Vol. 6. The Association for the Advancement of Artificial Intelligence Palo Alto.

[24] Luis Perez and Jason Wang. 2017. The Effectiveness of Data Augmentation in Image Classification using Deep Learning. *CoRR* abs/1712.04621 (2017).

[25] Akarsh Pokkunuru, Kalvik Jakkala, Arupjyoti Bhuyan, Pu Wang, and Zhi Sun. 2018. NeuralWave: Gait-Based User Identification Through Commodity WiFi and Deep Learning. In *IECON*. IEEE, 758–765.

[26] Kun Qian, Chenshu Wu, Zheng Yang, Yunhao Liu, and Kyle Jamieson. 2017. Widar: Decimeter-Level Passive Tracking via Velocity Monitoring with Commodity Wi-Fi. In *MobiHoc*. ACM, 6:1–6:10.

[27] Kun Qian, Chenshu Wu, Yi Zhang, Guidong Zhang, Zheng Yang, and Yunhao Liu. 2018. Widar2.0: Passive Human Tracking with a Single Wi-Fi Link. In *MobiSys*. ACM, 350–361.

[28] Xinping Rao and Zhi Li. 2019. MSDFL: a robust minimal hardware low-cost device-free WLAN localization system. *Neural Computing and Applications* 31, 12 (2019), 9261–9278. https://doi.org/10.1007/s00521-018-3945-8

[29] Tahsina Farah Sanam and Hana Godrich. 2019. A Multi-View Discriminant Learning Approach for Indoor Localization Using Bimodal Features of CSI. arXiv:cs.NI/1908.07370

[30] Souvik Sen, Dongho Kim, Stephane Laroche, Kyu-Han Kim, and Jeongkeun Lee. 2015. Bringing CUPID Indoor Positioning System to Practice. In *WWW*. ACM, 938–948.

[31] Souvik Sen, Bozidar Radunovic, Romit Roy Choudhury, and Tom Minka. 2012. You are facing the Mona Lisa: spot localization using PHY layer information. In *MobiSys*. ACM, 183–196.

[32] Siyu Shao, Pu Wang, and Ruqiang Yan. 2019. Generative adversarial networks for data augmentation in machine fault diagnosis. *Computers in Industry* 106 (2019), 85–93.

[33] Connor Shorten and Taghi M. Khoshgoftaar. 2019. A survey on Image Data Augmentation for Deep Learning. *J. Big Data* 6 (2019), 60.

[34] Wei Song, Liying Liu, Yifei Tian, Guodong Sun, Simon Fong, and Kyungeun Cho. 2017. A 3D localisation method in indoor environments for virtual reality applications. *HCIS* 7 (2017), 39.

[35] Zhuo Sun, Yiqiang Chen, Juan Qi, and Junfa Liu. 2008. Adaptive Localization through Transfer Learning in Indoor Wi-Fi Environment. In *ICMLA*. IEEE Computer Society, 331–336.

[36] Deepak Vasisht, Swarun Kumar, and Dina Katabi. 2015. Sub-Nanosecond Time of Flight on Commercial Wi-Fi Cards. In *SIGCOMM*. ACM, 121–122.

[37] Qinyong Wang, Hongzhi Yin, Hao Wang, Quoc Viet Hung Nguyen, Zi Huang, and Lizhen Cui. 2019. Enhancing Collaborative Filtering with Generative Augmentation. In *KDD*. ACM, 548–556.

[38] X. Wang, L. Gao, S. Mao, and S. Pandey. 2016. CSI-based Fingerprinting for Indoor Localization: A Deep Learning Approach. *IEEE Trans. Veh. Technol.* PP, 99 (2016), 1–1.

[39] Xiang Wang, Kai Wang, and Shiguo Lian. 2019. A Survey on Face Data Augmentation. *CoRR* abs/1904.11685 (2019).

[40] Kirti Wankhede, Bharati Wukkadada, and Vidhya Nadar. 2018. Just Walk-Out Technology and its Challenges: A Case of Amazon Go. In *2018 International Conference on Inventive Research in Computing Applications (ICIRCA)*. https://academic.microsoft.com/paper/2910638850

[41] Karl R. Weiss, Taghi M. Khoshgoftaar, and Dingding Wang. 2016. A survey of transfer learning. *J. Big Data* 3 (2016), 9.

[42] Wi-Fi Alliance. [n.d.]. Accurate indoor location with Wi-Fi connectivity. www.wi-fi.org/wi-fi-location.

[43] Di Wu, Boyu Wang, Doina Precup, and Benoit Boulet. 2019. Multiple Kernel Learning based Transfer Regression for Electric Load Forecasting. *IEEE Transactions on Smart Grid* (2019).

[44] Wei Xi, Jizhong Zhao, Xiang-Yang Li, Kun Zhao, Shaojie Tang, Xue Liu, and Zhiping Jiang. 2014. Electronic frog eye: Counting crowd using WiFi. In *INFOCOM*. IEEE, 361–369.

[45] Canwen Xu, Jing Li, Xiangyang Luo, Jiaxin Pei, Chenliang Li, and Donghong Ji. 2019. DLocRL: A Deep Learning Pipeline for Fine-Grained Location Recognition and Linking in Tweets. In *WWW*. ACM, 3391–3397.

[46] Dingqi Yang, Bingqing Qu, Jie Yang, and Philippe Cudré-Mauroux. 2019. Revisiting User Mobility and Social Relationships in LBSNs: A Hypergraph Embedding Approach. In *WWW*. ACM, 2147–2157.

[47] Zheng Yang, Chenshu Wu, and Yunhao Liu. 2012. Locating in fingerprint space: wireless indoor localization with little human intervention. In *MOBICOM*. ACM, 269–280.

[48] Xi Yin, Xiang Yu, Kihyuk Sohn, Xiaoming Liu, and Manmohan Chandraker. 2019. Feature Transfer Learning for Face Recognition with Under-Represented Data. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 5704–5713.

[49] Ruixiang Zhang, Tong Che, Zoubin Ghahramani, Yoshua Bengio, and Yangqiu Song. 2018. MetaGAN: An Adversarial Approach to Few-Shot Learning. In *NeurIPS*. 2371–2380.

[50] Yue Zheng, Yi Zhang, Kun Qian, Guidong Zhang, Yunhao Liu, Chenshu Wu, and Zheng Yang. 2019. Zero-Effort Cross-Domain Gesture Recognition with Wi-Fi. In *MobiSys*. ACM, 313–325.

[51] Fan Zhou, Xiaoli Yue, Goce Trajcevski, Ting Zhong, and Kunpeng Zhang. 2019. Context-aware Variational Trajectory Encoding and Human Mobility Inference. In *WWW*. ACM, 3469–3475.

[52] Bin Zou, Vasileios Lampos, and Ingemar J. Cox. 2019. Transfer Learning for Unsupervised Influenza-like Illness Models from Online Search Data. In *WWW*. ACM, 2505–2516.