Context-Aware Wireless-Based Cross-Domain Gesture Recognition

Hua Kang, Qian Zhang, *Fellow, IEEE*, and Qianyi Huang

***Abstract*—Recently, significant efforts have been made to enable WiFi-based gesture recognition. However, models trained with data collected from specific domain suffer from significant performance degradation when applied in a new domain. In practice, various WiFi sensing techniques have provided us with a full knowledge of domain information including discrete variables, i.e., environment and subject, as well as continuous variables, i.e., location and orientation. Previous works haven't fully explored these domain information or need to integrate substantial links' information to use them. Intuitively, we can boost gesture recognition accuracy by accounting for all these domain information with different properties. We propose a new framework not being restricted to link number which combines an adversarial learning scheme with feature disentanglement modules. They together conduct two-stage alignment between each of the source domains and the target domain to eliminate all gesture irrespective information. We also present an attention scheme based on discriminative information of each source and target domain to promote positive transfer from source to target domain. Our model is evaluated on the Widar 3.0 data set and achieves an improvement of 3%–12.7% in cross-domain average accuracy, demonstrating the superiority.**

***Index Terms*—Adversarial learning, context aware, cross-domain, gesture recognition.**

I. Introduction

HUMAN gesture recognition is a core part of human computer interaction, enabling multiple applications including smart home, health care and virtual reality. Previous attempts for gesture recognition utilize sensing modalities varying from visual [1]–[3], wearable [4], [5] to acoustics [6], [7]. However, they suffer from inherent drawbacks including privacy leakage, inconvenience as well as discomfort and limited sensing range. WiFi has emerged as a powerful sensing technique for gesture recognition due to its characteristics of privacy-protection, device-free and ubiquity. By deploying several wireless device pairs and analyzing the signal transmitted between them, we can infer people's gestures between the transmitter and receiver since different gestures lead to different time-varying transmission patterns of wireless signals. However, the signal arriving at the receiver not only contains information of the performed gesture but also carries substantial information of environment, subject, location and orientation of the subject. In specific, the signal propagates and interacts with the environment, undergoing penetration, reflection and diffraction, which are closely related to the surroundings. The subjects may have different motion amplitudes and speeds, and their body shapes also have an impact on the received signal. Different locations and orientations of the subject influence the amplitudes, changing patterns of the wireless metrics and have different blockage impacts on the signal. We use the term "domain" to summarize these factors uncorrelated with the gesture. Thus, the model trained with measurements in one domain often suffers from significant performance degradation when applied in a new domain. Consequently, labor-intensive data collection and labeling efforts are required for each deployment domain. However, there are infinite domains and it is impossible to cover all of them. This drastically restricts the generalization of WiFi sensing techniques.

Recent works have explored cross-domain WiFi-based gesture recognition models. For example, WiAG [8] proposes a translation model which can generate virtual samples in different configurations (i.e., location and orientation) using real samples in one configuration. In this way, they can achieve position and orientation agnostic gesture recognition. But they did not consider cases that across environments and subjects. EI [9] borrows the idea of adversarial domain adaptation and plays a minimax game between domain discriminator and feature extractor to align the distributions of source domain and target domain. However, they only consider discrete domain variables, i.e., environment and subject, and assign labels to them for the discriminator to distinguish. For continuous factors, i.e., location and orientation, it is impossible to assign a label for each domain. Widar 3.0 [10] generates a domain-invariant feature, body-coordinate velocity profile (BVP) to train a one-fits-all model. But they need as many as six links to calculate the BVP which is impractical in home scenarios, and the accuracy drops to lower than 70% when there are only two links.

Manuscript received November 16, 2020; revised January 29, 2021; accepted February 28, 2021. Date of publication March 11, 2021; date of current version August 24, 2021. This work was supported in part by the Key-Area Research and Development Program of Guangdong Province under Grant 2020B010164001; in part by the National Natural Science Foundation of China under Grant 62002150; in part by the Project of FANet: PCL Future Greater-Bay Area Network Facilities for Large-Scale Experiments and Applications under Grant LZC0019; in part by the Research Grants Council (RGC) of Hong Kong, China, under Contract CERG 16204418, Contract 16203719, Contract 16204820, and Contract R8015; and in part by the Guangdong Natural Science Foundation under Grant 2017A030312008. *(Corresponding author: Qian Zhang.)*

Hua Kang and Qian Zhang are with the Department of Computer Science and Engineering, Hong Kong University of Science and Technology, Hong Kong (e-mail: hkangae@cse.ust.hk; qianzh@cse.ust.hk).

Qianyi Huang is with the Institute of Future Networks, Southern University of Science and Technology, Shenzhen 518055, China, and also with the Network Communication Research Center, Peng Cheng Laboratory, Shenzhen 518066, China (e-mail: huangqy@sustech.edu.cn).

Digital Object Identifier 10.1109/JIOT.2021.3064890

Here, we consider a common and practical scenario in indoor environments where only two receivers are deployed for gesture recognition. The reason why we deploy two receivers is twofold. On the one hand, two links is a practical setting in places, e.g., home or office, which may not have many receivers. On the other hand, two links provide two views of the gesture which can compensate for the impact of occlusion to some extent. As at least three links are required to generate reliable BVP without ambiguity, with only two links, we can only use relatively primitive data, e.g., Doppler frequency shift (DFS) profile, as input. We collect gesture data from domains differing in settings of environment, subject, location and orientation. None of the existing methods except Widar 3.0 (uses BVP) [10] have taken all the four factors into consideration. They are only designed for transferring from labeled data to unlabeled data with differences in certain factors, e.g., environment and subject. As these four factors have different properties where environment and subject are discrete and countable while location and orientation are continuous and uncountable, they cannot be directly integrated into existing frameworks.

In this article, we apply wireless-based gesture recognition under the above mentioned situation with two links available in indoor environments. To relieve data labeling burden, we consider multisource unsupervised domain adaptation where there are multiple source domains with labeled data and a target domain with unlabeled data. Note that the target domain can be composed of multiple data distributions, i.e., domains, but we do not exactly know how the data are allocated to the data distributions. Through a comprehensive consideration of influential factors including discrete variables, i.e., environment and subject, and continuous variables, i.e., location and orientation, we present a deep learning model using an adversarial learning scheme together with feature disentanglement modules that can remove influences of these gesture irrelevant factors. Meanwhile, we use an attention scheme whose attention is based on source domains' discriminators' outputs to reflect different similarities between multiple source domains and the target domain to mitigate negative transfer.

Our main contribution is as follows.

1) We propose a novel deep learning framework using an adversarial network and a feature disentanglement part for multisource unsupervised domain adaptation considering both discrete domain factors such as environment and gesture performer and continuous domain factors such as location and orientation to fully discover the inner structure of data.
2) We consider differences between multiple source domains and the target domain and introduce an attention scheme to promote positive transfer between source and target domain.
3) We conduct comprehensive experiments on the Widar 3.0 data set [10] under a practical configuration where only two links' data are used. Our model achieves an average accuracy of 87.8%, 91.8%, 92.5%, 87.1%, and 85.7% across environments, subjects, locations, orientations and four influential factors, respectively, showing the superiority of the method with respect to classification accuracy.

II. Related Work

A. Transfer Learning

Traditional machine learning algorithms have an assumption that the train and test data share the same distribution. However, this may not hold in real applications. Transfer learning has emerged to address the problem by transferring the knowledge from some previous tasks to a target task with fewer data or without labels. Our work is related to domain adversarial training approaches [11], [12] and multisource unsupervised domain adaptation [13]–[15].

Domain Adversarial Training: With the development of GAN [16], adversarial training network becomes an important tool to improve performance. The generator is used to extract domain invariant features and is called feature extractor. The discriminator takes in the extracted features from source and target domain and tries to distinguish which domain they are from. The predictor is to predict the ground-truth labels. The total loss is the weighted sum of classification loss and domain discrimination loss. [17], [18] are the pioneer works on this area. Zhao *et al.* [19] proposed a conditional adversarial architecture which conditions on both extracted latent representation and discriminative information conveyed in the classifier predictions. Although the above architectures are effective, they treat each domain as a discrete label and cannot deal with situations where there are different types of domain information including both discrete variables and continuous variables.

Multisource Unsupervised Domain Adaptation: Most unsupervised domain adaptation methods consider single source *versus* single target. If multiple sources are available, domain shifts between source domains should be taken into consideration [14], [15], [20]. [21], [22] provide theoretical supports for multisource unsupervised domain adaptation problem. DCTN [13] based on distribution weighted combining rule proposed by [22], designs multiway adversarial learning to minimize discrepancy between each source domain and the target domain and multisource classifiers' predictions are integrated with perplexity scores to classify target samples. Our work is based on the architecture of DCTN while accounting for both discrete domain variables and continuous domain variables and designing an attention scheme in the training phase to promote positive transfer.

B. Cross-Domain Gesture Recognition

There are many prior works focusing on cross-domain gesture recognition to reduce data collection and labeling efforts and generalize the recognition model. There are two kinds of methods, making improvements from the model side especially seeking help from transfer learning and from the data side including generating virtual samples and extracting domain invariant features.

For those methods with the help of transfer learning, CrossSense [23] proposes an offline trained ANN-based roaming model mapping features from one environment to another.

But it does not fit to multiple source domains situations. EI [9] uses an adversarial training scheme together with several constraints to generalize the model to new environments and new subjects but it does not consider location and orientation's influence. ASTTL [24] considers cross data set wearable activity recognition problem by two steps: 1) source selection and 2) activity transfer. However, selecting only one source domain and adopting geodesic flow kernel (GFK) rather than deep learning models may limit the performance.

For those methods with the help of data, WiAG [8] only requires users to provide all the gestures at one configuration (i.e., location and orientation) and derives a translation function from one configuration to another to generate signal features for the target domain for model training. Widar 3.0 [10] extracts a domain invariant feature, BVP as input and builds a one-fits-all model with a combination of convolutional neural networks (CNNs) and gated recurrent units (GRUs). But at least three receivers are required to build BVP with low possibility of ambiguity.

Our article designs a new framework making use of all the domain information to boost classification accuracy of multi-source unsupervised domain adaptation under the case where there are only two receivers deployed in the environment.

III. Problem Setup

The input data contain both labeled data and unlabeled data. We refer to the data with and without gesture labels as source and target domain, respectively. All the data are attached with domain labels including four dimensions, i.e., environment, subject, location and orientation, indicating under what configurations the data are collected. Note that these domain information may be called attributes or features in other literatures and the domains are distinguished by these domain information. Thus, we may also call them domain for simplicity in the following text. For example, we may say that the source domain is subject 1 and target domain is subject 2. In this article, we consider a general and practical situation where the configurations for collecting labeled data are different from that for collecting unlabeled data and the settings for collecting labeled data are also different from each other. As environment and subject are discrete variables which can be represented as domain labels, we use these 2-D domain information to divide source labeled data into multiple domains, each with a discrete label. Suppose there are N such environment-subject pair divided source domains and these source domains correspond to N different underlying distributions $\{p_{s_i}(x, y)\}_{i=1}^{N}$. Source domains' data are sampled from N distributions, respectively, and have gesture labels, denoted as $(X_{s_i}, Y_{s_i}) = \{x_{s_i}^j, y_{s_i}^j\}_{j=1}^{n_{s_i}}$, where $i \in \{1, \ldots, N\}$. All the source domain data have the ground-truth location and orientation when performing the gesture, and they are continuous variables, denoted as $(L_{s_i}, O_{s_i}) = \{l_{s_i}^j, o_{s_i}^j\}_{j=1}^{n_{s_i}}$, where $i \in \{1, \ldots, N\}$. The orientation and location information of the person can be calculated by motion tracking approaches [8], [25], [26].

For the target domain, data can also be sampled from multiple underlying distributions, in other words, from multiple domains. But we do not explicitly divide them into multiple discrete domains according to environment-subject pairs like what we do for source domain. We simply regard the target domain as a whole set without label observation, denoted as $X_t = \{x_t^j\}_{j=1}^{n_t}$. The differences among target domain data instances will be explored by our scheme. And the target domain data also have the ground-truth location and orientation of the gesture, denoted as $(L_t, O_t) = \{l_t^j, o_t^j\}_{j=1}^{n_t}$. All the source domains and target domain have the same gesture categories.

IV. Methodology

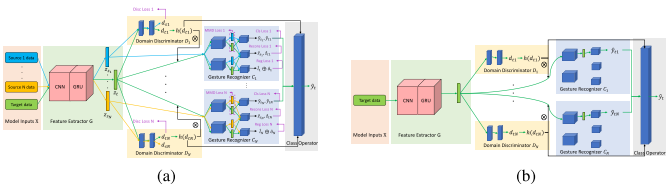
In this section, we present our two-stage adversarial domain adaptation framework for wireless gesture recognition. First, we provide an overview of the methodology. Next, we introduce model input formats. Then, we illustrate each module of the framework in turn. Finally, we give the objective and training method.

A. Overview

An overview of the model is shown in Fig. 1. To train a model which also works well in a new target domain, we first collect some unlabeled data from the target configuration and use these unlabeled target data together with the source data to train a model. Then for more data collected from the target domain, the model should still be effective. As the collected gesture data are time series data and there exist multiple links, the data after preprocessing may still be very complex. We adopt deep learning framework to derive discriminative features from these complex data. In the training phase, our input data are multiple source domains with label observations divided by environment-subject pairs and a target domain without label observations as shown in the left hand side of Fig. 1(a). In the test phase, we feed all the target data into the well-trained model. The target data only pass through some of the model components and the paths are shown by arrows in Fig. 1(b).

The input data are first transformed into low-dimensional representations $\mathbf{Z}$ by a common feature extractor which is composed of CNNs followed by GRUs. To mitigate domain discrepancies between target and each source, each source adopts a source-specific domain discriminator and a source-specific gesture recognizer. Each source-specific domain discriminator takes in representations $\mathbf{Z}$ from the corresponding source and the target, and labels them by $\mathbf{d}$ which represent the probability of $\mathbf{Z}$ coming from source domain. Thus, the goal of each domain discriminator is to maximize the domain label prediction accuracy which contradicts the goal of common feature extractor. By this adversarial scheme, feature extractor tries its best to extract domain label invariant features to cheat domain discriminators in which way target and each source can be aligned.

Each source-specific gesture recognizer first disentangles $\mathbf{Z}$ into *class-related features* $\mathbf{P}$ and *domain-related features* $\mathbf{Q}$. The former are further fed into source-specific gesture classifiers and calculate classification loss while the latter further pass through domain factor regressors and calculate regression loss. The above two losses together with maximum mean



(a) (b)

Fig. 1. Model overview. (a) Train. (b) Test.

discrepancy (MMD) loss between source *class-related features* and target *class-related features* as well as reconstruction loss between reconstructed $\hat{\mathbf{Z}}$ and original $\mathbf{Z}$ constitute gesture recognizer's loss.

To promote positive transfer between each source domain and the target domain, we introduce an attention scheme. Attentions are calculated by the outputs of source-specific domain discriminators, indicating that we enhance training of source domains which share more similarities with the target. The final classification results are the weighted sum of all the source-specific gesture recognizers' outputs where the weights are also calculated by domain discriminators' outputs.

In Fig. 1(b), for the test phase, target data only pass through partial modules of the whole framework. The target data are first fed into the well-trained common feature extractor and the extracted latent representations pass through N source-specific domain discriminators to calculate the attention values. Then, the attention-weighted representations pass through the source-specific class-related disentanglers and classifiers of N gesture recognizers to get N classification results. Finally, these results are combined according to attention values to get the final classification results.

B. Model Inputs

The proposed model can be utilized for various types of wireless data, such as DFS, channel state information (CSI) or BVP [10]. We only need to change the structure of our feature extractor to adapt to different data inputs. In this article, under the situation where only two links are deployed in the environment, we choose DFS as the input. Compared to BVP, DFS is more promising with only two links' data because BVP usually needs at least three links to overcome ambiguity as illustrated in [10]. In the step of transforming DFS to BVP in [10], each time instance is considered independently which incurs ambiguity since information along time axis is neglected. However, by directly feeding DFS into deep neural networks, it is possible to recover the gesture and eliminate ambiguity leveraging temporal information. Compared to CSI, DFS better reflects changes in velocity incurred by hand movements during performing the gestures. As it takes a certain amount of time T to perform a gesture, the collected data is a time sequence. We denote each time snapshot in spectrograms as a DFS profile. It is a matrix with dimension $F \times M$, where F is the number of sampling points in the frequency domain, and M is the transceiver link number. So the input DFS series are with dimension $T \times F \times M$. Based on such gesture sequences, we can derive gesture categories through careful design of network architecture to explore spatial temporal information of input data.

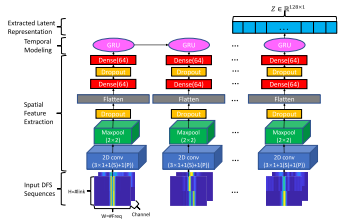


Fig. 2. Architecture of feature extractor G.

C. Feature Extractor

To reduce the parameter size and prevent overfitting, we use one common extractor G to extract representations from all the source domains and the target domain. We use an architecture similar to that in [10] which first extracts spatial features from each single DFS profile with CNNs and then models the temporal dependencies of the whole sequence with GRUs. We choose GRUs because they are capable of learning long-term representations and they achieve comparable performance while involving fewer parameters compared to long short-term memory (LSTM). The architecture of feature extractor is shown in Fig. 2. Let $\mathbf{X}$ represent the input from any source domain or target domain, i.e., $\mathbf{X} = \{\{X_{s_i}\}_{i=1}^{N}, X_t\}$. For the tth DFS profile with dimension $F \times M$, where $F = 121$ and $M = 2$, we adopt a (3×1) kernel, 1 stride and 1 padding in which way two links' data can be extracted independently in the first layer. After applying a max pooling layer to down-sample the two links' features and a dropout layer to reduce overfitting, the $2-D$ features are flattened into a vector. Then the vector is fed into two 64-units dense layers with Relu activation to get a higher level representation $\mathbf{V}_t$. To extract temporal relationships within DFS series, the outputs $\mathbf{V} = \{\mathbf{V}_t, t = 1, 2, \ldots, T\}$ are fed into single-layer GRUs and a 128-D feature vector is generated. We use $\mathbf{Z} = \{\{\mathbf{Z}_{s_i}\}_{i=1}^{N}, \mathbf{Z}_t\}$ to represent these extracted feature representations of all the source domains and the target domain. Let Θ_G be the set of CNNs and GRUs parameters. Given the input DFS series $\mathbf{X}$, we can express functional relationships between $\mathbf{Z}$ and $\mathbf{X}$ as follows:

$$\mathbf{Z} = G(\mathbf{X}; \Theta_G). \tag{1}$$

The extracted features $\mathbf{Z}$ are expected to be domain invariant which can be realized by adversarial learning.

D. Domain Discriminator

We build N domain discriminators like [13] to distinguish each source domain and the target domain, denoted as $D = \{D_{s_i}\}_{i=1}^{N}$, where the ith source domain has domain discriminator D_{s_i}. Each source-specific domain discriminator has the same architecture, i.e., two fully connected layers where the first layer has leaky Relu activation while the second layer has sigmoid activation to get the output. The target latent representations $\mathbf{Z}_t$ flow to all the source-specific domain discriminators and the ith source representations $\mathbf{Z}_{s_i}$ only trigger the ith source-specific domain discriminator D_{s_i}.

Let $\Theta_D^{s_i}$ be the set of source ith domain discriminator's parameters. Given the latent representations $\mathbf{Z}$, we can express the functional relationship between domain discriminator's outputs $\mathbf{d} = \{\mathbf{d}_{s_i}, \mathbf{d}_{ti}\}_{i=1}^{N}$ and $\mathbf{Z} = \{\{\mathbf{Z}_{s_i}\}_{i=1}^{N}, \mathbf{Z}_t\}$ as follows:

$$\mathbf{d}_{ti} = D_{s_i}\left(\mathbf{Z}_t; \Theta_D^{s_i}\right), \quad \mathbf{d}_{s_i} = D_{s_i}\left(\mathbf{Z}_{s_i}; \Theta_D^{s_i}\right) \tag{2}$$

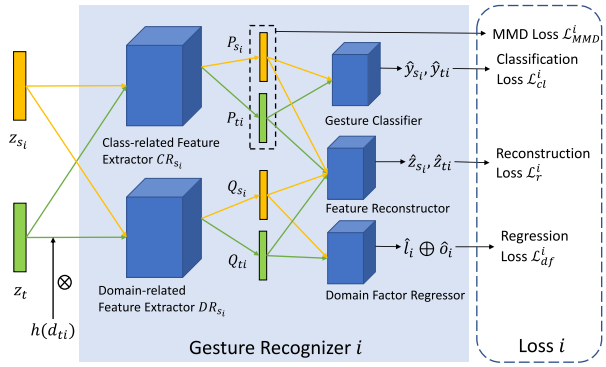


Fig. 3. Components of gesture recognizer i.

where $i \in \{1, 2, \ldots, N\}$, $\mathbf{d}_{s_i}$ are the ith source-specific domain discriminator's outputs for source domain i's data and $\mathbf{d}_{ti}$ are the ith source-specific domain discriminator's outputs for target data. The target data outputs of domain discriminators are denoted as $\mathbf{d}_t = \{\{d^j_{ti}\}^{n_t}_{j=1}\}^N_{i=1}$, where each $d^j_{ti} \in [0, 1]$. The outputs of domain discriminators reflect the similarities between the target domain and the source domain. If the output is close to 0.5, it means that the domain discriminator is uncertain about whether the sample is from source or target. So the target shares more similarities with the source domain. On the contrary, if the output is close to 0 or 1, there may be large discrepancy between source and the target. These outputs are used in Section IV-F for weighting the features.

The goal of each D_{s_i} is to identify whether the input originates from the ith source or the target. The loss of D_{s_i} denoted as $\mathcal{L}^i_d$ is defined as follows:

$$\mathcal{L}^i_d = -\frac{1}{n_{s_i}} \sum_{j=1}^{n_{s_i}} \log\left(\mathbf{d}^j_{s_i}\right) - \frac{1}{n_t} \sum_{j=1}^{n_t} \log\left(1 - \mathbf{d}^j_{ti}\right). \tag{3}$$

The total discriminator loss $\mathcal{L}_d$ is the mean of all the domain discriminators' losses

$$\mathcal{L}_d = \frac{1}{N} \sum_{i=1}^{N} \mathcal{L}^i_d. \tag{4}$$

E. Gesture Recognizer

The gesture recognizer C is composed of N source-specific gesture recognizers $\{C_{s_i}\}^N_{i=1}$. Each source-specific gesture recognizer independently predicts the gesture labels for the target domain and it contains the following components: class-related feature extractor, domain-related feature extractor, gesture classifier, domain factor regressor and feature reconstructor. The detailed architecture of gesture recognizer is shown in Fig. 3 which is an enlargement of the diagram of gesture recognizer i in Fig. 1. We then introduce each component one by one.

1) Feature Disentanglement: We want the representations fed into gesture classifiers to be domain-invariant. Suppose we have the perfect domain discriminators such that the extracted features $\mathbf{Z}$ should be invariant to the domain labels, i.e., pairs of environment and subject. However, there are other domain related features affecting the extracted features and further affecting the classification results, i.e., location and orientation, which are explicitly known through existing sophisticated passive tracking systems, e.g., LiFS [27] , IndoTrack [26] and Widar2.0 [25]. We want to further disentangle the extracted features into *class-related features* and *domain-related features* where *class-related features* are responsible for gesture classification and *domain-related features* are responsible for domain information regression. For each source domain i, we design CR_{s_i} to extract *class-related features* and DR_{s_i} to extract *domain-related features*. Only source i's features and target features pass through CR_{s_i} and DR_{s_i}. Both CR_{s_i} and DR_{s_i} are of the same architecture consisting of a dropout layer followed by a fully connected layer with batch normalization and Relu activation. The outputs are 64 dimensional vectors. Let $\mathbf{P} = \{\mathbf{P}_{s_i}, \mathbf{P}_{ti}\}^N_{i=1}$ represent *class-related features* and $\mathbf{Q} = \{\mathbf{Q}_{s_i}, \mathbf{Q}_{ti}\}^N_{i=1}$ represent *domain-related features*. The functional relationships between $\mathbf{P}$, $\mathbf{Q}$ and common extractor extracted features $\mathbf{Z}$ are as follows:

$$\mathbf{P}_{s_i} = \mathrm{CR}_{s_i}\left(\mathbf{Z}_{s_i}; \Theta^{s_i}_{\mathrm{CR}}\right), \quad \mathbf{Q}_{s_i} = \mathrm{DR}_{s_i}\left(\mathbf{Z}_{s_i}; \Theta^{s_i}_{\mathrm{DR}}\right) \tag{5}$$

$$\mathbf{P}_{ti} = \mathrm{CR}_{s_i}\left(\mathbf{Z}_t; \Theta^{s_i}_{\mathrm{CR}}\right), \quad \mathbf{Q}_{ti} = \mathrm{DR}_{s_i}\left(\mathbf{Z}_t; \Theta^{s_i}_{\mathrm{DR}}\right) \tag{6}$$

where $i \in \{1, 2, \ldots, N\}$, $\Theta^{s_i}_{\mathrm{CR}}$ are the parameters of source i's class-related feature disentangler, $\Theta^{s_i}_{\mathrm{DR}}$ are the parameters of source i's domain-related feature disentangler, $\mathbf{P}_{s_i}$ is the class-related feature of source i, $\mathbf{P}_{ti}$ are the *class-related features* of the target calculated by the ith source-specific class-related feature disentangler, $\mathbf{Q}_{s_i}$ are the *domain-related features* of source i and $\mathbf{Q}_{ti}$ are the *class-related features* of the target calculated by the ith source-specific domain-related feature disentangler. The extracted $\mathbf{P}$ and $\mathbf{Q}$ are responsible for gesture classification and domain factor regression, respectively.

2) Gesture Classifier: We have N gesture classifiers $\mathrm{CL} = \{\mathrm{CL}_{s_i}\}^N_{i=1}$ corresponding to each source domain. The extracted *class-related features* flow to gesture classifier which is composed of one fully connected layer with Softmax activation. The outputs $\hat{\mathbf{y}} = \{\hat{\mathbf{y}}_{s_i}, \hat{\mathbf{y}}_{ti}\}^N_{i=1}$ are calculated by

$$\hat{\mathbf{y}}_{s_i} = \mathrm{CL}_{s_i}\left(\mathbf{P}_{s_i}; \Theta^{s_i}_{\mathrm{CL}}\right), \quad \hat{\mathbf{y}}_{ti} = \mathrm{CL}_{s_i}\left(\mathbf{P}_{ti}; \Theta^{s_i}_{\mathrm{CL}}\right). \tag{7}$$

The purpose of each gesture classifier is to correctly classify gesture categories of input data. For source domain with label, we calculate the cross entropy loss

$$\mathcal{L}^{s_i}_{\mathrm{cl}} = -\frac{1}{n_{s_i}} \sum_{j=1}^{n_{s_i}} \mathbf{y}^{j\top}_{s_i} \log \hat{\mathbf{y}}^j_{s_i}. \tag{8}$$

For target domain without label, if the confidence of one class overpasses a threshold γ, we assign the sample pseudo label $\widetilde{\mathbf{y}}_{ti}$ to supervise the training, the way to calculate $\widetilde{\mathbf{y}}_{ti}$ follows the method introduced in Section IV-F which integrates multiple classifiers' outputs. For target samples not overpassing the threshold, following [11], we minimize the conditional entropy with respect to the target distribution to improve target sample prediction confidence, so the classification loss of the ith source-specific classifier for target domain is as follows:

$$\mathcal{L}^{ti}_{\mathrm{cl}} = \frac{1}{n_t} \sum_{j=1}^{n_t} \left[\boldsymbol{b}^j_{ti} \widetilde{\mathbf{y}}^{j\top}_{ti} \log \hat{\mathbf{y}}^j_{ti} + \left(1 - \boldsymbol{b}^j_{ti}\right) \hat{\mathbf{y}}^{j\top}_{ti} \log \hat{\mathbf{y}}^j_{ti}\right] \tag{9}$$

where $\boldsymbol{b}^j_{ti} = \mathrm{sign}[(\max(\hat{\mathbf{y}}^j_{s_i} - \gamma, 0))_+]$, meaning that when output $\hat{\mathbf{y}}^j_{ti}$ overpasses threshold γ, $\boldsymbol{b}^j_{ti}$ equals to 1, otherwise,

$\boldsymbol{b}_{ti}^{j}$ equals 0. Based on (8) and (9), the gesture classification loss is the weighted sum of source and target classification losses

$$\mathcal{L}_{\text{cl}} = \frac{1}{N}\sum_{i=1}^{N}\left(\mathcal{L}_{\text{cl}}^{s_i} + \alpha\mathcal{L}_{\text{cl}}^{ti}\right) \tag{10}$$

where α is the weighting parameter. From (10), we can observe that the source domain data only influence the corresponding source-specific gesture classifier and the target domain data, though without label, have impact on all the gesture classifiers.

3) Domain Factor Regressor: The extracted *domain-related features* are fed into source-specific domain factor regressors to calculate the regression loss with respect to the ground-truth location and orientation. We denote source-specific domain factor regressors as $\text{DF} = \{\text{DF}_{s_i}\}_{i=1}^{N}$. Through the supervision of regression loss, we can make sure that the input disentangled features are domain-related thus do not influence gesture classification task. The architecture is a fully-connected layer. The estimated orientations $\hat{\boldsymbol{o}} = \{\hat{\boldsymbol{o}}_{s_i}, \hat{\boldsymbol{o}}_{ti}\}_{i=1}^{N}$ and locations $\hat{\boldsymbol{l}} = \{\hat{\boldsymbol{l}}_{s_i}, \hat{\boldsymbol{l}}_{ti}\}_{i=1}^{N}$ are calculated by

$$\hat{\boldsymbol{l}}_{\tau} \oplus \hat{\boldsymbol{o}}_{\tau} = \text{DF}_{s_i}\left(\boldsymbol{Q}_{\tau}; \Theta_{\text{DF}}^{s_i}\right) \tag{11}$$

where $\tau \in \{s_i, ti\}$ and $\hat{\boldsymbol{l}}_{\tau} \oplus \hat{\boldsymbol{o}}_{\tau}$ indicates that the output is a vector concatenating the predicted location and orientation.

As the data set does not provide the accurate orientation, from [8] we know that a change in orientation of up to 45° does not have a significant impact on gesture recognition accuracy. We term orientation estimation as a classification problem. So we use cross entropy loss for orientation estimation and mean-squared error loss (MSE) for location estimation. The loss function for each domain factor regressor and the whole loss of domain factor regressors are as follows:

$$\begin{aligned}\mathcal{L}_{\text{df}}^{i} = {} & \frac{1}{n_{s_i}}\sum_{j=1}^{n_{s_i}}\left[\left(\boldsymbol{l}_{s_i}^{j} - \hat{\boldsymbol{l}}_{s_i}^{j}\right)^2 - \boldsymbol{o}_{s_i}^{j\top}\log\text{Softmax}\left(\hat{\boldsymbol{o}}_{s_i}^{j}\right)\right] \\ & + \frac{1}{n_t}\sum_{j=1}^{n_t}\left[\left(\boldsymbol{l}_{t}^{j} - \hat{\boldsymbol{l}}_{ti}^{j}\right)^2 - \boldsymbol{o}_{t}^{j\top}\log\text{Softmax}\left(\hat{\boldsymbol{o}}_{ti}^{j}\right)\right]\end{aligned} \tag{12}$$

$$\mathcal{L}_{\text{df}} = \frac{1}{N}\sum_{i=1}^{N}\mathcal{L}_{\text{df}}^{i}. \tag{13}$$

4) Feature Reconstructor: To make sure that the extracted *class-related features* and *domain-related features* can recover the original feature, they are concatenated and fed into reconstructor $R = \{R_{s_i}\}_{i=1}^{N}$ which is composed of a fully connected layer. The estimated latent features $\hat{\boldsymbol{Z}} = \{\hat{\boldsymbol{Z}}_{s_i}, \hat{\boldsymbol{Z}}_{ti}\}_{i=1}^{N}$ are calculated by

$$\hat{\boldsymbol{Z}}_{s_i} = R_{s_i}\left(\boldsymbol{P}_{s_i} \oplus \boldsymbol{Q}_{s_i}; \Theta_{R}^{s_i}\right), \quad \hat{\boldsymbol{Z}}_{ti} = R_{s_i}\left(\boldsymbol{P}_{ti} \oplus \boldsymbol{Q}_{ti}; \Theta_{R}^{s_i}\right) \tag{14}$$

where $i \in \{1, 2, \ldots, N\}$ and $\oplus$ represents concatenation. The MSE is calculated between the recovered feature and the original feature as follows:

$$\mathcal{L}_{r}^{i} = \frac{1}{n_{s_i}}\sum_{j=1}^{n_{s_i}}\left(\boldsymbol{Z}_{s_i}^{j} - \hat{\boldsymbol{Z}}_{s_i}^{j}\right)^2 + \frac{1}{n_t}\sum_{j=1}^{n_t}\left(\boldsymbol{Z}_{t}^{j} - \hat{\boldsymbol{Z}}_{ti}^{j}\right)^2 \tag{15}$$

$$\mathcal{L}_{r} = \frac{1}{N}\sum_{i=1}^{N}\mathcal{L}_{r}^{i}. \tag{16}$$

5) MMD Loss: MMD loss is commonly used in transfer learning [12] as a standard distribution distance metric. It can be used to learn a representation that minimizes the distance between the source and target distributions. We calculate the MMD loss between each source and target domain's *class-related feature* to align their distributions before classification. The MMD loss is formulated as follows:

$$\mathcal{L}_{\text{MMD}}^{i} = \left\| \frac{1}{n_{s_i}}\sum_{j=1}^{n_{s_i}}\phi\left(\boldsymbol{P}_{s_i}^{j}\right) - \frac{1}{n_t}\sum_{j=1}^{n_t}\phi\left(\boldsymbol{P}_{ti}^{j}\right)\right\|_{\mathcal{H}}^{2} \tag{17}$$

where $\phi(\cdot)$ is the feature map which maps the original representations into the reproducing kernel Hilbert space (RKHS) $\mathcal{H}$ endowed with some characteristic kernel k where $k(\boldsymbol{P}_{s_i}^{j}, \boldsymbol{P}_{ti}^{j}) = <\phi(\boldsymbol{P}_{s_i}^{j}), \phi(\boldsymbol{P}_{ti}^{j})>$ and $<\cdot,\cdot>$ represents inner product. The whole MMD loss is the average of each source domain's MMD loss

$$\mathcal{L}_{\text{MMD}} = \frac{1}{N}\sum_{i=1}^{N}\mathcal{L}_{\text{MMD}}^{i}. \tag{18}$$

F. Domain Attention and Class Operator

In our architecture so far, each target sample would pass through each source domain's gesture recognizer including those with few similarities. As the target domain may contain multiple latent domains, for instance, the target domain contains multiple subjects' data. In this way, each target sample has different similarities with each source domain. As we have a common feature extractor, if we want to align each source domain with the target domain, source domains that are not similar to the target distribution will weaken the closeness of features between the target and its nearest source domain. So intuitively, we should enhance the influence of similar source domains while decline the impact of dissimilar source domains.

Specifically, for each target sample j, we assign a domain attention vector $a^j \in \mathbf{R}^{N\times 1}$, where N is the number of source domains. The ith element a_i^j is the probability of assigning the target sample j to the ith source domain. a_i^j reflects the similarity between the target sample j and the source domain i. We utilize the output $\boldsymbol{d}_{ti}^{j}$ of domain discriminator D_{s_i} to calculate a_i^j, and it is formulated as

$$a_i^j = g \circ f\left(\boldsymbol{d}_{ti}^{j}\right) \tag{19}$$

where $f(x) = 1 - 2|x - 0.5|$ and $g(x) = \begin{cases} x, & \text{if} \quad x \geq \epsilon \\ 0, & \text{if} \quad x < \epsilon, \end{cases}$ where ϵ is a threshold. This indicates that if the ith discriminator is too confident (output is close to 0) to identify the origin of the target sample, the weight of it will be very small or equal to 0, meaning that the parameter update for this sample ignores the ith gesture recognizer's influence.

As the output of each module of gesture recognizer can be written as functions of $\boldsymbol{Z_t}$ and $\boldsymbol{Z_{s_i}}$, all the target losses can be written as functions of $\boldsymbol{Z_t}$ and $\boldsymbol{Z_{s_i}}$. We use l_{τ}^{i} representing the loss functions of the ith gesture recognizer with latent representations $\boldsymbol{Z_{s_i}}, \boldsymbol{Z_t}$ as input where $\tau \in \{\text{cl}, \text{df}, r, \text{MMD}\}$. Let $\Theta_{C}^{s_i}$ be the parameters of the ith gesture recognizer, thus

we have the above loss functions summarized as

$$\mathcal{L}_\tau^i = l_\tau^i\left(\boldsymbol{Z}_{\boldsymbol{s_i}}, \boldsymbol{Z_t}; \Theta_C^{s_i}\right). \tag{20}$$

Let $\boldsymbol{a_i} = [a_i^1, a_i^2, \ldots, a_i^{n_t}]^\top \in \mathbf{R}^{n_t \times 1}$ be the attention scores of assigning target samples to the ith source domain. We use a_i^j multiplying $\boldsymbol{Z_t^j}$ extracted by G and then feed them into the ith source-specific gesture recognizer, the loss can be reformulated as

$$\mathcal{L}_\tau^i = l_\tau^i\left(\boldsymbol{Z}_{\boldsymbol{s_i}}, \boldsymbol{a_i} \odot \boldsymbol{Z_t}; \Theta_C^{s_i}\right) \tag{21}$$

where $\boldsymbol{a_i} \odot \boldsymbol{Z_t}$ is the dot product of the extracted features and the attention vector. Here, the attention scheme has the similar function as dropout, to enhance the influence of similar source domains' gesture recognizers and decline the influence of dissimilar source domains' gesture recognizers. Thus, the attention scheme measures the similarities between the extracted source domain representations and target domain representations and enables similar source domains' classification ability to transfer. $\mathcal{L}_\tau = (1/N)\sum_{i=1}^N \mathcal{L}_\tau^i$, where $\tau \in \{\text{cl}, \text{df}, r, \text{MMD}\}$ meaning that gesture classification loss, domain factor regression loss, reconstruction loss and MMD loss obey the above formulas.

The last step before we get target samples' categories is to integrate multiple classification results via a class operator. As we have calculated each target sample's domain attention vector, we can use the normalized vector to weigh the classification result of each source domain. The jth target sample's prediction is calculated as follows:

$$y_t^j = \sum_{i=1}^N \frac{a_i^j}{\sum_{i=1}^N a_i^j} \hat{\boldsymbol{y}}_{ti}^j \tag{22}$$

where a_i^j is the attention of assigning sample j to source domain i and $\hat{\boldsymbol{y}}_{ti}^j$ is the output of ith source's gesture classifier for target sample j.

G. Objective and Training

The final objective is composed of domain discriminator loss, MMD loss, classification loss, domain regression loss and reconstruction loss. The objective is as follows:

$$\min_{G,C}\max_{D} \mathcal{L} = \mathcal{L}_{\text{cl}} - \beta\mathcal{L}_d + \eta\mathcal{L}_{\text{df}} + \rho\mathcal{L}_r + \xi\mathcal{L}_{\text{MMD}} \tag{23}$$

where β, η, ρ, ξ are weighting parameters, $\mathcal{L}_{\text{cl}}$ is the gesture classification loss, $\mathcal{L}_d$ is the domain discrimination loss, $\mathcal{L}_{\text{df}}$ is the domain factor regression loss, $\mathcal{L}_r$ is the feature reconstruction loss and $\mathcal{L}_{\text{MMD}}$ is the MMD loss. From (23), we can observe that feature extractor G tries its best to cheat domain discriminator D by maximizing $\mathcal{L}_d$ and at the same time promotes the performance of gesture recognizer C by minimizing losses from gesture recognizer, i.e., $\mathcal{L}_{\text{cl}}$, $\mathcal{L}_{\text{df}}$, $\mathcal{L}_r$ and $\mathcal{L}_{\text{MMD}}$.

Equation (23) can be divided into two parts, namely, gesture recognition loss, $\mathcal{L}_{\text{cls}}$, and adversarial loss, $\mathcal{L}_{\text{adv}}$, (23) can be written as

$$\min_{G,C}\max_{D} \mathcal{L} = \mathcal{L}_{\text{cls}} + \mathcal{L}_{\text{adv}} \tag{24}$$

where $\mathcal{L}_{\text{cls}} = \mathcal{L}_{\text{cl}} + \eta\mathcal{L}_{\text{df}} + \rho\mathcal{L}_r + \xi\mathcal{L}_{\text{MMD}}$ and $\mathcal{L}_{\text{adv}} = -\beta\mathcal{L}_d$. As there are multiple source domains and a target domain, when we minimize the above equation, the distributions change simultaneously, which leads to an oscillation that spoils our feature extractor. Following [13], we use domain confusion which performs stably to learn the mapping G. We have the following multidomain confusion loss:

$$\mathcal{L}'_{\text{adv}} = \frac{1}{N}\sum_{i=1}^N \left[\frac{1}{n_{s_i}}\sum_{j=1}^{n_{s_i}} \mathcal{L}_{cf}\left(\boldsymbol{d}_{s_i}^j\right) + \frac{1}{n_t}\sum_{j=1}^{n_t} \mathcal{L}_{cf}\left(\boldsymbol{d}_{ti}^j\right)\right] \tag{25}$$

where

$$\mathcal{L}_{cf}(x) = \frac{1}{2}\log(x) + \frac{1}{2}\log(1-x). \tag{26}$$

The training process adopts an alternative way to train three modules as shown in Algorithm 1.

Algorithm 1 Learning Algorithm

Input:
N source labeled datasets $\{X_{s_i}, Y_{s_i}\}_{i=1}^N$; target unlabeled dataset X_t; initiated feature extractor G, gesture recognizer C and domain discriminator D; confidence threshold γ; classification epochs T_c; extractor epochs T_g; discriminator interval T_d; weighting parameters $\alpha, \beta, \eta, \rho, \xi$.

Output:
well-trained feature extractor G^*, domain discriminator D^* and gesture recognizer C^*.

while not converged **do**
  for $epoch = 1 : T_g$ **do**
    Sample mini-batch from $\{X_{s_i}\}_{i=1}^N$ and X_t
    if mod(epoch, T_d)==0 **then**
      Update D by Eq. (23);
    end if
    Estimate discriminator outputs $\boldsymbol{d}_{ti}$, calculate attention $\boldsymbol{a_i}$, $i \in \{1, 2, .., N\}$ by Eq. (19).
    Calculate losses by Eq. (21).
    Update G by minimizing $\beta\mathcal{L}'_{adv} + \mathcal{L}_{cl} + \eta\mathcal{L}_{df}$;
  end for
  Estimate discriminator outputs $\boldsymbol{d}_{ti}$, calculate attention $\boldsymbol{a}_i$, $i \in \{1, 2, ..., N\}$ by Eq. (19). Estimate confidence for X_t by Eq. (22). Assign pseudo labels for X_t with confidence larger than γ.
  for epoch=1:T_c **do**
    Sample mini-batch from $\{X_{s_i}\}_{i=1}^N$ and X_t.
    Estimate discriminator outputs $\boldsymbol{d}_{ti}$, calculate attention $\boldsymbol{a_i}$, $i \in \{1, 2, ..., N\}$ by Eq. (19).
    Calculate $\mathcal{L}_{cls}$ by Eq. (21).
    Update G and C by $\mathcal{L}_{cls}$ from Eq. (24).
  end for
end while
return $G^* = G$, $C^* = C$, $D^* = D$

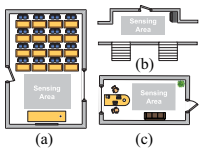


Fig. 4. Layouts of three evaluation environments, figure from [10]. (a) Classroom. (b) Hall. (C) office.

V. Experiments

This section presents the implementation and detailed performance of our framework.

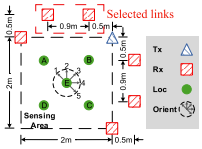


Fig. 5. Typical setup of devices and domains in one environment, figure modified from [10].

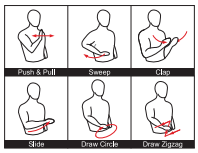


Fig. 6. Sketches of gestures evaluated in the experiment, figure from [10].

A. Experiment Methods

1) Data Set: We use the public Widar 3.0 [10] data set. The data set is collected from three rooms, 16 users, five locations, and five orientations in total. Fig. 4 shows the layouts and sensing areas of the three experimental environments containing various types, i.e., classroom, hall and office. Fig. 5 shows a typical example of the deployment of devices and domain configurations in the sensing area, which is a 2 m $\times$ 2 m square. As illustrated in the original paper, there are six receivers deployed in the environment, in our experiments, we consider a more practical situation where only the first two links are used for gesture recognition which are circled by the red dashed box as shown in Fig. 5. In this study, we classify six commonly used gestures as shown in Fig. 6. The data set contains 11 250 data samples (15 users $\times$ 5 positions $\times$ 5 orientations $\times$ 6 gestures $\times$ 5 instances).

2) Input and Preprocessing: We adopt DFS as our model input. As for the SignFi [28] baseline, we use denoised CSI as input. The DFS sequence has the shape of M (the number of receivers) $\times$ F (the number of Doppler frequency samples) $\times$ T (the number of time samples). As we first use CNN model to extract spatial information and then apply GRUs to extract temporal information, we treat each time snapshot as a DFS profile with dimension 2×121 and feed them into CNN model. The problem with it is that the sampling rate of DFS data is 1000 samples/s, so there are more than 1000 time steps in a DFS sequence. With so many time steps, the training of GRUs would suffer from gradient vanishing problem. Therefore, we reshape the DFS sequence as $M \times F \times C \times \lfloor (T/C) \rfloor$, where $M = 2, F = 121$ and $C = 100$. The above operation integrates 100 time steps and regards them as the channel number. In this way, the GRUs only contain around a dozen time steps and both GRUs and CNNs can be well trained.

B. Baseline Methods

1) CNNGRU Model: We modified the CNNGRU model used in [10] whose input is BVP a little bit to adapt to our DFS input. We make the architecture exactly the same as a combination of feature extractor G, a class-related feature extractor CR_i and a gesture classifier CL_i in our model. In this way, we can have a fair comparison between this baseline and our model by guaranteeing the same backbone architecture. We only use the source data to train and directly apply the model on the target data for prediction.

2) SignFi [28]: Apart from using DFS as input, we also show the CSI-based gesture recognition method's performance. SignFi is designed for sign language recognition. In our experiment, we modify the number of classes to 6 and use the same CNN architecture. The inputs are the concatenation of CSI magnitudes and phases of two links.

3) DCTN [13]: This article considers the situation where the labeled data are collected from diverse domains and the model is to be deployed in a single target domain. In this model, the domain is defined by discrete environment-subject pairs without considering differences caused by continuous locations and orientations. To make a fair comparison, the feature extractor and all the domain discriminators have the same architecture as our model. Each source domain's gesture classifier has the same architecture as the combination of our class-related feature extractor CR_i and gesture classifier CL_i since this is the path data passing through to get the gesture label.

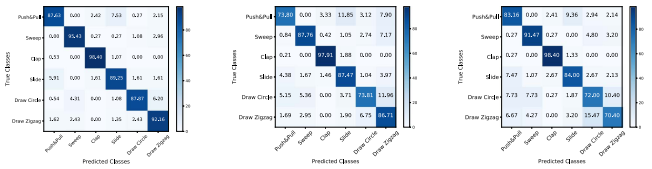
4) EI [9]: EI is a state-of-the-art adversarial learning scheme for unsupervised domain adaptation and it can be applied for multiple source domains and target domains. There is only one feature extractor, one domain discriminator and one gesture classifier in EI. The domain discriminator is responsible for identifying the correct domain label for all the data which is a multiclass classification problem. The feature extractor adopts the same architecture as that in our model. For domain discriminator, we only change the number of output units of the last layer to the number of all the domains and change the activation function from Sigmoid to Softmax. We also change the input units of domain discriminator to the dimension of the concatenation of gesture classifier's output and feature extractor's output. For gesture classifier, we use the same architecture as the combination of class-related feature extractor CR_i and gesture classifier CL_i of our model.

5) Widar 3.0 [10]: In Widar 3.0 paper, the authors extract the domain invariant feature BVP based on 6 links' DFS data and feed them into a model combining CNNs and GRUs. We use the method in that paper as a baseline with BVP as input. Note that this method needs 6 links' data while our model only uses 2 links' data. We will show that our model can achieve comparable results with less data under various configurations as Widar 3.0.

6) Variant of Our Model: As there are multiple modules in our architecture, how to make use of all the source domains to maximize benefits is essential. We substitute the domain attention scheme and obtain a variant of our model. Specifically, we do not use our domain discriminators' instance level outputs to amplify or reduce training of each category classifier. We use the same scheme as in [13] which uses target sample's loss to get weighted predictions and uses them to supervise the training of feature extractor. We also adopt the same class operator as that in [13].

TABLE I
ACROSS PERSON CLASSIFICATION ACCURACY (%) COMPARISON

Models	15,16→ 14	14,16→ 15	14,15→ 16	Avg	Gain
CNNGRU	78.3	82.6	79.4	80.1	0
SignFi	67.1	63.1	66.3	65.5	-14.6
EI	80.1	84.4	84.7	83.1	+3.0
DCTN	82.9	84.7	86.7	84.8	+4.7
Widar 3.0 (use 6 links)	84.7	86.1	88.9	86.6	+6.5
Our model (variant)	89.1	**92.1**	91.2	90.8	+10.7
Our model	**90.2**	91.1	**94.1**	**91.8**	**+11.7**

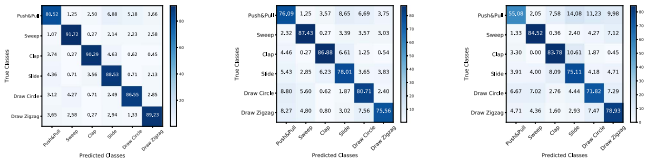


(a) (b) (c)

Fig. 7. Across person confusion matrix. (a) Our model ($P = 0.92, R = 0.92, F_1 = 0.92$). (b) DCTN ($P = 0.85, R = 0.84, F_1 = 0.84$). (c) EI ($P = 0.84, R = 0.84, F_1 = 0.84$).

TABLE II
ACROSS ROOM CLASSIFICATION ACCURACY (%) COMPARISON

Models	$R_2, R_3 \rightarrow R_1$	$R_1, R_3 \rightarrow R_2$	$R_1, R_2 \rightarrow R_3$	Avg	Gain
CNNGRU	61.1	71.4	76.0	69.5	0
SignFi	64.6	56.3	56.1	58.3	-11.2
EI	71.7	79.5	79.2	76.8	+7.3
DCTN	78.8	81.4	82.3	80.8	+11.3
Widar 3.0 (use 6 links)	**90.1**	**89.0**	86.4	**88.5**	**+19.0**
Our model (variant)	86.0	86.1	87.3	86.5	+17.0
Our model	87.3	87.4	**88.8**	87.8	+18.3



(a) (b) (c)

Fig. 8. Across room confusion matrix. (a) Our model ($P = 0.88, R = 0.88, F_1 = 0.88$). (b) DCTN ($P = 0.81, R = 0.81, F_1 = 0.81$). (c) EI ($P = 0.76, R = 0.76, F_1 = 0.75$).

C. Cross-Domain Evaluation

We evaluate the overall performance of our model on cases across different domain factors, including subject, environment, location and orientation. When we evaluate on each domain factor, we keep the other domain factors as the same. We also evaluate the overall performance of our model when all the four influential domain factors are different. For each case, 80% of the target domain data are used to train with the source data and all the target domain data are used to test, indicating that for more data collected from the same configuration, we do not need to train again. We conduct experiments using all the baseline methods to validate our model's superiority.

1) Across Subject: Different subjects' data may show differences due to subjects' different behaviors of performing gestures and their body shapes. To evaluate on different subjects, we use data from Room 1 where there are 8 people in total. From some preliminary experiments, we found that discrepancies between different subjects are relatively small compared to other domain factors. Without loss of generality, we choose a subset of $User_{14}$, $User_{15}$ and $User_{16}$ in Room 1 to evaluate the model where $User_{14}$ is a female and $User_{15}$ and $User_{16}$ are males. Note that we will evaluate on influence of different subject numbers in Section V-D3. Each time, we leave one user as the target domain and the other two users as the source domains. The results can be seen in Table I.

From Table I, we can observe that our model achieves the highest accuracy under three cases compared to other baseline models. Note that this accuracy is comparable to the result shown in Widar 3.0 [10] using seven person's data to train with six links. We can see from the table that our model outperforms Widar 3.0 when only using two persons' data to train, implying that our model is suitable for situations where fewer user data are available. We then compare confusion matrices of three domain adaptation methods, i.e., EI, DCTN and our model. The confusion matrices of three across person cases are integrated to get the average confusion matrix for our model, DCTN and EI as shown in Fig. 7(a)–(c), respectively. We can observe that for three models, the gesture "Push&Pull" always corresponds to a lower accuracy. As said in [10], the gesture is mostly performed just in front of the torso and is most likely to be blocked when conducted under orientations away from the receiver. By horizontal comparison with other models in each gesture, our model achieves the best results in all categories and has the highest average value in precision, recall and macro-F_1 score.

2) Across Room: In the data set, different rooms have different user numbers, to avoid data set size's influences, we keep each room's person number as the same. We choose 3 persons from each room and obtain a subset of the whole data set which is $User_{14}$, $User_{15}$ and $User_{16}$ in Room 1, $User_1$, $User_2$ and $User_6$ in Room 2 and $User_7$, $User_8$, and $User_9$ in Room 3. The experiments take one room's data as the target domain and the other two rooms' data as the source domains. As the subjects do not conduct the same experiments in all the rooms but only occur in one room, the across room case is actually across person at the same time. The experiment results are shown in Table II.

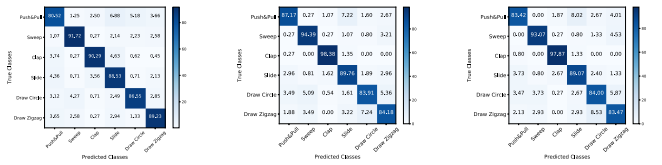
From Table II, we find that our model shows consistent results under several cases with an average accuracy of 87.8%. We can find that the amount of accuracy improvement is more than that in across person case compared with most baselines. Our model achieves similar accuracy as Widar 3.0 using 6 links.

From Fig. 8, we find that our model has the consistent and highest accuracy in each category while the other methods suffer from much confusion for certain gestures, e.g., clap and slide. Our model also remains the highest value of average precision, recall and macro F_1 score.

3) Across Location: To evaluate performance of across location situation, we use every combination of four locations, all five orientations, three users in room 1 as the source domain, and the last unseen location with the same set of orientations, users and rooms' data as the target domain. The result comparison and detailed per class accuracy are shown in Table III and Fig. 9, respectively. From Table III, we can find the average accuracy is 93.6%. The distinctions among the five cases are relatively small and the improvement of

TABLE III
ACROSS LOCATION CLASSIFICATION ACCURACY (%) COMPARISON

Models	$B,C,D,E \rightarrow A$	$A,C,D,E \rightarrow B$	$A,B,D,E \rightarrow C$	$A,B,C,E \rightarrow D$	$A,B,C,D \rightarrow E$	Avg	Gain
CNNGRU	83.9	87.5	86.0	79.6	88.0	85.0	0
SignFi	65.6	64.2	67.8	68.9	70.1	67.4	-17.6
EI	87.9	90.1	86.8	85.8	93.1	88.7	+3.7
DCTN	90.0	88.6	87.4	87.7	93.8	89.5	+4.5
Widar 3.0 (use 6 links)	87.7	87.1	**93.0**	**88.7**	93.4	90.0	+5.0
Our model (variant)	**92.9**	**96.4**	91.3	87.7	95.1	**92.7**	**+7.7**
Our model	91.5	95.8	91.7	87.5	**95.8**	92.5	+7.5

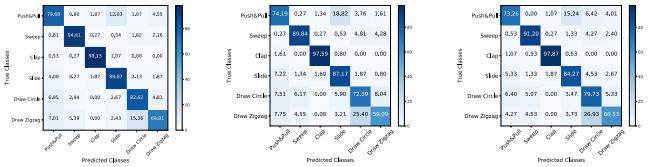


(a) (b) (c)

Fig. 9. Across location confusion matrix. (a) Our model (P=0.93,R=0.92,$F_1 = 0.92$). (b) DCTN ($P = 0.90, R = 0.90, F_1 = 0.90$). (c) EI (P=0.88,$R = 0.88, F_1 = 0.88$).

TABLE IV
ACROSS ORIENTATION CLASSIFICATION ACCURACY (%) COMPARISON

Models	$2,3,4,5 \rightarrow 1$	$1,3,4,5 \rightarrow 2$	$1,2,4,5 \rightarrow 3$	$1,2,3,5 \rightarrow 4$	$1,2,3,4 \rightarrow 5$	Avg	Gain
CNNGRU	60.5	71.9	75.0	81.3	67.9	71.3	0
SignFi	68.4	62.2	70.4	75.1	66.0	68.4	-2.9
EI	70.4	81.1	88.1	90.8	70.7	80.2	+8.9
DCTN	71.5	80.9	87.5	92.0	74.2	81.2	+9.9
Widar 3.0 (use 6 links)	**82.4**	87.5	90.2	88.7	**83.4**	86.4	+15.1
Our model (variant)	73.0	85.3	88.8	94.8	82.2	84.7	+13.4
Our model	82.1	**88.6**	**92.6**	**94.9**	77.2	**87.1**	**+15.8**



(a) (b) (c)

Fig. 10. Across orientation confusion matrix. (a) Our model ($P = 0.86, R = 0.86, F_1 = 0.86$). (b) DCTN ($P = 0.80, R = 0.80, F_1 = 0.80$). (c) EI ($P = 0.82, R = 0.81, F_1 = 0.81$).

our model compared to other methods using DFS varies from 2.5% to 7.5%.

We also noted that our model has the similar average accuracy as the variant which substitutes our attention scheme. The reason why the improvement is not obvious under across location case is that our attention scheme is based on the outputs of our source-specific domain discriminators whose outputs should be similar in this situation as the source contains the same subjects. For example, we use location 5 as the target, the number of discriminators is determined by the number of environment-subject pairs so there are three discriminators in total. The ith discriminator needs to distinguish between data from $User_i$ collected at location 1 to 4 and data from $\{User_{j_1}, User_i, User_{j_2}\}$ collected at location 5. If a target sample is from $User_i$, then it has the most in common with source i and assigns the most weight to that domain. As each user collected the same quantity of data, the weight for each source domain should be similar. So the gain of attention scheme is not clear.

From Fig. 9, our model still remains best results in all the categories and the highest average precision, recall and macro F_1 score compared to other domain adaptation methods.

4) Across Orientation: Similar to the across location settings, we divide Room 1's *User*14, *User*15 and *User*16's data by orientation and let each orientation as the target and the other four orientations as the source. Table IV shows the overall trend of accuracy variation, 2, 3, 4 orientations as the middle orientations get higher accuracy than those at edges. We have an overall accuracy of 87.1%. Orientations 3 and 4 have an accuracy above 90% while the accuracy declines over 10% for orientations 1 and 5. This is reasonable since orientation 1 and 5 are too biased and the receiver may not capture the full pattern of gestures. Widar 3.0 using BVP has better results than our model in these orientations, since BVP is generated from 6 links' DFS profiles, a more completed picture. And the integrated confusion matrix in Fig. 10 shows that Push&Pull, "Draw circle" and "Draw zigzag" gestures have a lower accuracy which may be due to body blockage interference influence where the other three gestures get a high accuracy all above 85%. Compared to other methods, our model achieves the highest average precision, recall and macro F_1 value.

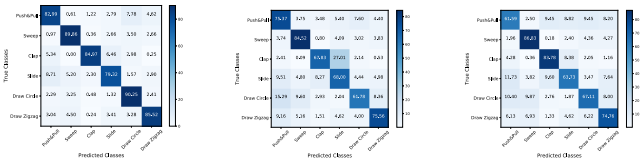
TABLE V
CASES TESTED ACROSS FOUR FACTORS

Case No.	Source domain				Target domain			
	Room	Subject	Location	Orientation	Room	Subject	Location	Orientation
Case 1	1,2	14,15,16,1,2,6	1,2,3,4	1,2,3,5	3	7,8,9	1,2,3,4,5	1,2,3,4,5
Case 2	1,3	14,15,16,7,8,9	1,2,3,4	1,2,3,5	2	1,2,6	1,2,3,4,5	1,2,3,4,5
Case 3	2,3	1,2,6,7,8,9	1,2,3,4	1,2,3,5	1	14,15,16	1,2,3,4,5	1,2,3,4,5

TABLE VI
ACROSS FOUR INFLUENTIAL FACTORS CLASSIFICATION ACCURACY (%) COMPARISON

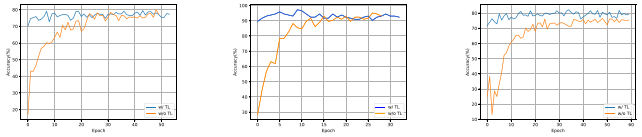
Models	Case 1	Case 2	Case 3	Avg	Gain
CNNGRU	74.5	74.3	57.2	68.7	0
SignFi	65.6	58.9	64.4	63.0	-5.7
EI	76.2	76.9	65.9	73.0	+4.3
DCTN	81.0	77.7	56.5	71.9	+3.2
Widar 3.0 (use 6 links)	83.7	**86.6**	**87.7**	**86.0**	**+17.3**
Our model (variant)	**86.1**	84.7	73.7	81.5	+12.8
Our model	**86.1**	84.6	86.1	85.7	+17.0

5) Across Four Influential Factors: To further validate the superiority of our proposed framework, we evaluate a challenging case where all the four factors are different between the target and all the source domains. Since there are too many combinations of the four factors (i.e., environment, subject, location and orientation), we choose three cases to test. Table V shows the corresponding source and target domains of each case. Based on across room cases in Section V-C2, each time we still use one room as the target and the other two rooms as the source domains. Here, each room contains three persons which is consistent with the settings in Section V-C2 For orientation and location, this time we only use data from four locations and four orientations in the source rooms and test on data from all the five locations and orientations in the target room. Table VI shows the results. We find that our model has consistent results under several cases with an average accuracy of 85.7%. From the perspective of average accuracy under three cases. Compared with the Widar 3.0 method that needs six links, we achieve the same average accuracy.



(a) (b) (c)

Fig. 11. Across four influential factors confusion matrix. (a) Our model ($P = 0.86, R = 0.85, F_1 = 0.86$). (b) DCTN ($P = 0.73, R = 0.72, F_1 = 0.72$). (c) EI ($P = 0.73, R = 0.73, F_1 = 0.73$).



(a) (b) (c)

Fig. 12. Accuracy comparison of model w/ or w/o pretrained weights. (a) Source: user 14, 15, and 16 in room 1; target: user 7, 8, and 9 in room 3; pretrained weights from case, where user 1, 2, and 6 in room 2 are the target. (b) Source: user 14, 15, and 16 in room 1; target: user 13 in room 1; pretrained weights from case where user 12 in room 1 is the target. (c) Source: user 5, 10, and 11 in room 1; target: user 7, 8, and 9 in room 3; pretrained weights from case, where user 14, 15, and 16 in room 1 are the source and user 1, 2, and 6 in room 2 are the target.

From Fig. 11, we find that our model has the consistent and highest accuracy in each category while the other methods suffer from much confusion for certain gestures, e.g., draw circle. Our method also achieves the highest value of precision, recall and macro F_1 score.

6) Training for New Target Domain: To relieve training efforts for new target domain, we can use previous trained model to initialize the weights for new models. In this way, fewer epochs can be used to make the model converge. Fig. 12 shows the accuracy comparison between model trained with previous model weights and model trained from scratch. We can see that for all the cases, compared with models trained from scratch, the accuracy of model trained with previous model weights is higher and the epochs to reach the stable accuracy is much fewer.

7) Run-Time Analysis: We report the training and inference time of our proposed method, EI and DCTN. We run the experiments on a cluster node with 2 Intel Xeon E5-2670 v3 and 2 Nvidia Tesla K80. We measure each epoch's time and the number of epochs needed to converge to show the total training time. We evaluate under across person case where user 14 and 15 are the source domains and user 16 is the target domain. Fig. 13 shows the convergence curve of the three methods. We can see that our method quickly converges within 20 epochs while the other two methods have similar convergence speed and cost, about 100 epochs. Table VII shows the training time and inference time. The target domain have 750 samples in total and the total inference time is 32 s, and thus it takes about 42.7 ms to infer each sample's class.

D. Parameter Study

In this section, we tune the input parameters of our framework to see their impact. We will conduct evaluations on impact of selected links, impact of location and orientation estimation error, impact of subject number, impact of link number, impact of location number and impact of orientation number.

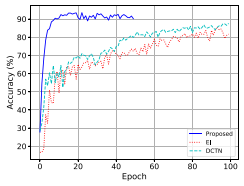


Fig. 13. Convergence

TABLE VII
RUNNING TIME (IN SECONDS)

Method	Train time per epoch	Epoch number	Train time	Inference time
Our model	1220.0	13	15860.0	32.0
EI	564.3	93	52479.9	62.5
DCTN	360.4	98	35319.2	49.6

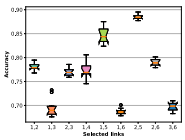


Fig. 14. Impact of selected links.

1) Impact of Selected Links: This section studies the impact of selecting different receiver locations. According to the deployment of receivers in Fig. 5, we select nine different combinations of two links neglecting symmetrical settings and conduct experiments under across orientation situation where orientation 5 is the target. We choose to evaluate on across orientation case since its performance is closely correlated with link locations as inappropriate link locations cannot cover the entire sensing area and capture human movements. Fig. 14 shows the accuracy using data of different links. We can find that there are large performance discrepancies between different selected links because we can only use such a limited number of links and the locations of them are especially important for gesture recognition in oblique orientations with respect to receivers. According to the results, we can achieve a higher accuracy when we deploy receivers from both sides of the transmitter to maximize the coverage of the sensing area and various orientations. And the accuracy even achieves 88.5% accuracy with as large as 10% accuracy improvement compared to our aforementioned settings.

2) Impact of Location and Orientation Estimation Error: Locations and orientations as inputs of our framework are estimated by WiFi-based motion tracking systems and inevitably have errors. To evaluate their impact, we add uniform noise to the ground-truth locations and orientations. For locations, we add noise uniformly distributed in the range $[-\delta_l, \delta_l]$ to each coordinate, where $\delta_l \in \{0, 0.2, 0.4, 0.6\}$. For orientations, as we term orientation estimation as classification problem and there are 45° difference between different categories. So if we add noise whose absolute value is less than 22.5°, the categories will not be changed and the accuracy will not be influenced. Thus, we only add noise uniformly distributed in the range $[-\delta_o, \delta_o]$, where $\delta_o \in \{0, 30, 40\}$. We evaluate impact of location error on across location case where location

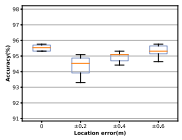


Fig. 15. Impact of location error.

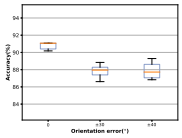


Fig. 16. Impact of orientation error.

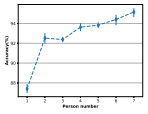


Fig. 17. Impact of subject number.

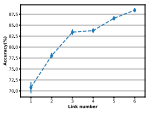


Fig. 18. Impact of link number.

5 is the target while evaluating impact of orientation error on across orientation case where orientation 3 is the target. As shown in Fig. 15, the overall accuracy remains above 94% when the range of location error is within ±0.6 m, indicating that the model is not sensitive to location noise. We can observe from Fig. 16 that decreases in accuracy as orientation error increases is larger than deviation observed for location error. But the accuracy drops when the orientation noise range is as high as ±30° and the decrease is approximately 3%. Therefore, we believe that our model is robust enough within error range of estimated locations and orientations.

3) Impact of Subject Number: In this section, we study the impact of number of subjects in the training set. In specific, we fix *User*16 in Room 1 as the target, and sequentially enlarge the training set from only *User*15 to all the other experiment users in Room 1. There are 7 cases in total and the results are shown in Fig. 17. The gesture recognition accuracy increases from 87.4% to 95.2% as the number of subjects varies from 1 to 7. The reason comes from that the increase in the amount of training data allows our model to be trained well. Note that the accuracy discrepancy is less than 10% and the accuracy remains above 87% when there is only one person in the training set. Also note that there are slight fluctuations in the middle as the number of persons increases which owe to the added person carrying little net and valid information.

4) Impact of Link Number: There are 6 links' data provided in the data set. We will study how much gain in accuracy can be obtained by increasing link number in this section. We consider a case that has low performance, that is across orientation situation where orientation 5 is regarded as target. From Fig. 5, we find that the layout is symmetrical along the diagonal axis and our selected receivers are at the same side. So by

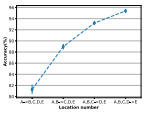


Fig. 19. Impact of location number.

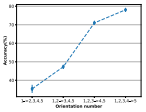


Fig. 20. Impact of orientation number.

increasing link number, we can get a more complete record of the signal. We sequentially add the receivers into the selected collections from receiver 1 to receiver 6 and the results are shown in Fig. 18. As we increase the link number, the accuracy gradually increases and it reaches above 87% when we use all the six links. It can be seen that using four links has a similar accuracy as using three links. Different architectures of feature extractor G contribute to this as different link number forms profiles with different shapes to be fed into the CNN model. By carefully adapting the parameters, such as kernel size, of different link numbers, it is expected to achieve better results. There are also works [29] considering how to integrate different views' data.

5) Impact of Location Number: We explore whether using fewer locations as source can still get good result. We design four cases where there are 4, 3, 2, 1 locations in the source domain while the remaining locations are in the target domain. Fig. 19 shows that the accuracy declines over 10% as the number of source locations decreases from 4 to 1. But when there is only 1 location as source, the accuracy is still above 81% showing robustness of our model for different locations. In this way, when we collect labeled source data, we can conduct gestures in fewer locations which significantly reduces data collection burdens.

6) Impact of Orientation Number: Similar to study on impact of location number, we arrange the 5 orientations in sequence and separate them in the middle to form four cases where the training data set has 1, 2, 3, 4 orientations, respectively. The results are shown in Fig. 20. We observe that the accuracy rapidly drops from 78.1% to 35.4% as the orientation number varies from 4 to 1. The reason is twofold. On the one hand, as the number of orientation decreases, the size of training data set decreases. On the other hand, there is large discrepancy between different orientations due to possible blockages and different reflection angles. To have a reasonable performance, we need to keep at least three orientations to get an accuracy above 70%. And the users should try to face the transceivers when performing gestures to increase recognition accuracy without excessive angles.

VI. Conclusion

In this article, we present a deep learning framework to recognize device free human gestures with only two links' signal data. Especially, the labeled data are collected from multiple

domains which are different from where the unlabeled data are collected. With a thorough consideration of gesture category irrespective domain information including environment, subject, location and orientation, the proposed model contains an adversarial training scheme together with feature disentanglement modules to remove all these domain information's influences. An attention scheme whose attention reflects different similarities between source domains and target samples and a class operator integrating each source classifier's output are proposed to promote positive transfer from source to target. Extensive experiments on the Widar 3.0 data set demonstrate the superiority of the proposed framework.

REFERENCES

[1] S. S. Rautaray and A. Agrawal, "Vision based hand gesture recognition for human computer interaction: A survey," *Artif. Intell. Rev.*, vol. 43, no. 1, pp. 1–54, 2015.

[2] P. Narayana, J. R. Beveridge, and B. A. Draper, "Gesture recognition: Focus on the hands," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Salt Lake City, UT, USA, 2018, pp. 5235–5244.

[3] N. Quader, J. Lu, P. Dai, and W. Li, "Towards efficient coarse-to-fine networks for action and gesture recognition," in *Proc. Eur. Conf. Comput. Vis.*, 2020, pp. 35–51.

[4] S. Jiang *et al.*, "Feasibility of wrist-worn, real-time hand, and surface gesture recognition via sEMG and IMU sensing," *IEEE Trans. Ind. Informat.*, vol. 14, no. 8, pp. 3376–3385, Aug. 2018.

[5] H. Truong *et al.*, "CapBand: Battery-free successive capacitance sensing wristband for hand gesture recognition," in *Proc. 16th ACM Conf. Embedded Netw. Sens. Syst.*, 2018, pp. 54–67.

[6] R. Nandakumar, A. Takakuwa, T. Kohno, and S. Gollakota, "CovertBand: Activity information leakage using music," *Proc. ACM Interact. Mobile Wearable Ubiquitous Technol.*, vol. 1, no. 3, pp. 1–24, 2017.

[7] Y. Wang, J. Shen, and Y. Zheng, "Push the limit of acoustic gesture recognition," *IEEE Trans. Mobile Comput.*, early access, Oct. 19, 2020, doi: 10.1109/TMC.2020.3032278.

[8] A. Virmani and M. Shahzad, "Position and orientation agnostic gesture recognition using WiFi," in *Proc. 15th Annu. Int. Conf. Mobile Syst. Appl. Serv.*, 2017, pp. 252–264.

[9] W. Jiang *et al.*, "Towards environment independent device free human activity recognition," in *Proc. 24th Annu. Int. Conf. Mobile Comput. Netw.*, 2018, pp. 289–304.

[10] Y. Zheng *et al.*, "Zero-effort cross-domain gesture recognition with Wi-Fi," in *Proc. 17th Annu. Int. Conf. Mobile Syst. Appl. Serv.*, 2019, pp. 313–325.

[11] R. Shu, H. H. Bui, H. Narui, and S. Ermon, "A DIRT-T approach to unsupervised domain adaptation," 2018. [Online]. Available: arXiv:1802.08735

[12] E. Tzeng, J. Hoffman, N. Zhang, K. Saenko, and T. Darrell, "Deep domain confusion: Maximizing for domain invariance," 2014. [Online]. Available: arXiv:1412.3474

[13] R. Xu, Z. Chen, W. Zuo, J. Yan, and L. Lin, "Deep cocktail network: Multi-source unsupervised domain adaptation with category shift," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Salt Lake City, UT, USA, 2018, pp. 3964–3973.

[14] M. Mancini, L. Porzi, S. R. Bulò, B. Caputo, and E. Ricci, "Boosting domain adaptation by discovering latent domains," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Salt Lake City, UT, USA, 2018, pp. 3771–3780.

[15] X. Peng, Q. Bai, X. Xia, Z. Huang, K. Saenko, and B. Wang, "Moment matching for multi-source domain adaptation," in *Proc. IEEE Int. Conf. Comput. Vis.*, Seoul, South Korea, 2019, pp. 1406–1415.

[16] I. Goodfellow *et al.*, "Generative adversarial nets," in *Advances in Neural Information Processing Systems*. Red Hook, NY, USA: 2014, pp. 2672–2680.

[17] Y. Ganin *et al.*, "Domain-adversarial training of neural networks," *J. Mach. Learn. Res.*, vol. 17, no. 1, pp. 2096–2030, 2016.

[18] Z. Cao, M. Long, J. Wang, and M. I. Jordan, "Partial transfer learning with selective adversarial networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Salt Lake City, UT, USA, 2018, pp. 2724–2732.

[19] M. Zhao, S. Yue, D. Katabi, T. S. Jaakkola, and M. T. Bianchi, "Learning sleep stages from radio signals: A conditional adversarial architecture," in *Proc. 34th Int. Conf. Mach. Learn.*, vol. 70, 2017, pp. 4100–4109. [Online]. Available: JMLR.org

[20] Z. Pei, Z. Cao, M. Long, and J. Wang, "Multi-adversarial domain adaptation," 2018. [Online]. Available: https://arxiv.org/abs/1809.02176

[21] J. Blitzer, K. Crammer, A. Kulesza, F. Pereira, and J. Wortman, "Learning bounds for domain adaptation," in *Advances in Neural Information Processing Systems*. Vancouver, BC, Canada: Curran Associates, Inc., 2008, pp. 129–136.

[22] Y. Mansour, M. Mohri, and A. Rostamizadeh, "Domain adaptation with multiple sources," in *Advances in Neural Information Processing Systems*. Red Hook, NY, USA: Curran, 2009, pp. 1041–1048.

[23] J. Zhang, Z. Tang, M. Li, D. Fang, P. Nurmi, and Z. Wang, "CrossSense: Towards cross-site and large-scale WiFi sensing," in *Proc. 24th Annu. Int. Conf. Mobile Comput. Netw.*, 2018, pp. 305–320.

[24] X. Qin, Y. Chen, J. Wang, and C. Yu, "Cross-dataset activity recognition via adaptive spatial-temporal transfer learning," *Proc. ACM Interact. Mobile Wearable Ubiquitous Technol.*, vol. 3, no. 4, pp. 1–25, 2019.

[25] K. Qian, C. Wu, Y. Zhang, G. Zhang, Z. Yang, and Y. Liu, "Widar2.0: Passive human tracking with a single Wi-Fi link," in *Proc. 16th Annu. Int. Conf. Mobile Syst. Appl. Serv.*, 2018, pp. 350–361.

[26] X. Li *et al.*, "IndoTrack: Device-free indoor human tracking with commodity Wi-Fi," *Proc. ACM Interact. Mobile Wearable Ubiquitous Technol.*, vol. 1, no. 3, pp. 1–22, 2017.

[27] J. Wang *et al.*, "LiFs: Low human-effort, device-free localization with fine-grained subcarrier information," in *Proc. 22nd Annu. Int. Conf. Mobile Comput. Netw.*, 2016, pp. 243–256.

[28] Y. Ma, G. Zhou, S. Wang, H. Zhao, and W. Jung, "SignFi: Sign language recognition using WiFi," *Proc. ACM Interact. Mobile Wearable Ubiquitous Technol.*, vol. 2, no. 1, pp. 1–21, 2018.

[29] H. Xue *et al.*, "DeepMV: Multi-view deep learning for device-free human activity recognition," *Proc. ACM Interact. Mobile Wearable Ubiquitous Technol.*, vol. 4, no. 1, pp. 1–26, 2020.

Hua Kang received the bachelor's degree in transportation engineering from Tongji University, Shanghai, China, in July 2018. She is currently pursuing the Ph.D. degree with the Department of Computer Science and Engineering, Hong Kong University of Science and Technology, Hong Kong.

Her research interests include Internet of Things, wireless sensing, and deep learning.

Qian Zhang (Fellow, IEEE) received the B.S., M.S., and Ph.D. degrees in computer science from Wuhan University, Wuhan, China, in 1994, 1996, and 1999, respectively.

In September 2005, she joined Hong Kong University of Science and Technology (HKUST), Hong Kong, where she is currently a Tencent Professor of Engineering and the Chair Professor with the Department of Computer Science and Engineering. She is also serving as the Co-Director of Huawei-HKUST Innovation Lab and the Director of Digital Life Research Center, HKUST. Before that, she was with Microsoft Research Asia, Beijing, China, in July 1999, where she was the Research Manager of the Wireless and Networking Group. Her current research interests include Internet of Things, smart health, mobile computing and sensing, and wireless networking, as well as cyber security.

Prof. Zhang is currently serving as the Editor-in-Chief for the IEEE TRANSACTIONS ON MOBILE COMPUTING. She was a Members-at-Large of the IEEE Communications Society from 2016 to 2018.

Qianyi Huang received the bachelor's degree in computer science from Shanghai Jiao Tong University, Shanghai, China, in 2013, and the Ph.D. degree from the Department of Computer Science and Engineering, Hong Kong University of Science and Technology, Hong Kong, in 2018.

She is with the Institute of Future Networks, Southern University of Science and Technology, Shenzhen, China, and also with the Network Communication Research Center, Peng Cheng Laboratory, Shenzhen. She has published a number of papers in top-ranking journals and conferences, including IEEE/ACM TRANSACTIONS ON NETWORKING, IEEE TRANSACTIONS ON MOBILE COMPUTING, MobiCom, UbiComp, and INFOCOM. Her research interests lie in mobile computing, Internet of Things, and security.