CrossGR: Accurate and Low-cost Cross-target Gesture Recognition Using Wi-Fi

XINYI LI, Northwest University; Northwest University IoT Research Center, China
LIQIONG CHANG*, Northwest University; International Joint Research Centre for the Battery-Free IoT, China
FANGFANG SONG, Northwest University; Northwest University IoT Research Center, China
JU WANG, Northwest University IoT Research Center, China
XIAOJIANG CHEN, Northwest University; International Joint Research Centre for the Battery-Free IoT, China
ZHANYONG TANG, Northwest University; Northwest University - Jingdong Wisdom Cloud Joint Research Center for AI & IoT, China
ZHENG WANG, School of Computing, University of Leeds, United Kingdom

This paper focuses on a fundamental question in Wi-Fi-based gesture recognition: “Can we use the knowledge learned from some users to perform gesture recognition for others?”. This problem is also known as cross-target recognition. It arises in many practical deployments of Wi-Fi-based gesture recognition where it is prohibitively expensive to collect training data from every single user. We present CrossGR, a low-cost cross-target gesture recognition system. As a departure from existing approaches, CrossGR does not require prior knowledge (such as who is currently performing a gesture) of the target user. Instead, CrossGR employs a deep neural network to extract user-agnostic but gesture-related Wi-Fi signal characteristics to perform gesture recognition. To provide sufficient training data to build an effective deep learning model, CrossGR employs a generative adversarial network to automatically generate many synthetic training data from a small set of real-world examples collected from a small number of users. Such a strategy allows CrossGR to minimize the user involvement and the associated cost in collecting training examples for building an accurate gesture recognition system. We evaluate CrossGR by applying it to perform gesture recognition across 10 users and 15 gestures. Experimental results show that CrossGR achieves an accuracy of over 82.6% (up to 99.75%). We demonstrate that CrossGR delivers comparable recognition accuracy, but uses an order of magnitude less training samples collected from the end-users when compared to state-of-the-art recognition systems.

CCS Concepts: • **Human-centered computing → Ubiquitous and mobile computing**.

Additional Key Words and Phrases: Cross-target, Wireless Sensing, Gesture Recognition, Wi-Fi, Deep Learning

*This is the corresponding author.

Authors’ addresses: Xinyi Li, Northwest University; Northwest University IoT Research Center, China, clq.nwu.edu.cn; Liqiong Chang, Northwest University; International Joint Research Centre for the Battery-Free IoT, China, clq.nwu.edu.cn; Fangfang Song, Northwest University; Northwest University IoT Research Center, China, songfang@stumail.nwu.edu.cn; Ju Wang, Northwest University IoT Research Center, China, wangju@nwu.edu.cn; Xiaojiang Chen, Northwest University; International Joint Research Centre for the Battery-Free IoT, China, xjc@nwu.edu.cn; Zhanyong Tang, Northwest University; Northwest University - Jingdong Wisdom Cloud Joint Research Center for AI & IoT, China, zytang@nwu.edu.cn; Zheng Wang, School of Computing, University of Leeds, United Kingdom , z.wang5@leeds.ac.uk.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.
© 2021 Association for Computing Machinery.
2474-9567/2021/3-ART21 $15.00
https://doi.org/10.1145/3448100

ACM Reference Format:
Xinyi Li, Liqiong Chang, Fangfang Song, Ju Wang, Xiaojiang Chen, Zhanyong Tang, and Zheng Wang. 2021. CrossGR: Accurate and Low-cost Cross-target Gesture Recognition Using Wi-Fi. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.* 5, 1, Article 21 (March 2021), 23 pages. https://doi.org/10.1145/3448100

1 INTRODUCTION

Human gesture and activity recognition underpins many real-world applications like health care [3, 7], fitness tracking [12, 19] and safety authentication [38, 47]. Traditionally, gesture recognition was performed using wearable sensors or devices like a smartphone or smartwatch. Recent studies have shown that it is possible to achieve device-free activity recognition by monitoring how human activities affect the wireless signal transmission path. This leads to a burgeoning research field of wireless sensing [4, 7, 20]. Among all existing device-free gesture recognition systems, the Wi-Fi-based system is attractive due to the widespread deployment of Wi-Fi devices [28]. It is particularly useful for smart spaces as it can be easily deployed using as few as two wireless routers (i.e., low-cost), and being less privacy intrusive than other infrastructure-based solutions such as video monitoring [51].

Numerous Wi-Fi based gesture recognition systems have been proposed [28, 34, 35]. Unfortunately, existing approaches have a fundamental drawback by requiring a labour-intensive and time-consuming process of collecting training measurements to characterize how wireless channel metrics - such as channel state information (CSI) or received signal strength indicator (RSSI) - are affected by each target gesture performed by each user. The reason for requiring this expensive training data collection process is clear - the wireless signal characteristics (e.g., CSI or RSSI) correlate to not only the human gestures but also the user's physical features (e.g., heights, weights, and body shape) and behavior habits. However, while it may be feasible to collect such data from each occupant of a home, asking each employee or visitor to provide training measurements in a smart office setting is impractical. This drawback limits the scale at which Wi-Fi based gesture recognition can be operated.

Ideally, we would like to build a system from training data collected from a few users, and the learned system can be applied to many other new users while still achieve high recognition accuracy. Doing so can greatly reduce the cost and user burden for training data collection, increasing the scale and uptake of Wi-Fi-based gesture sensing. This open research problem is known as *cross-target recognition.*

Some of the most recent studies have attempted to utilize adversarial learning [18, 45] to suppress the effect of user-specific factors in gesture and activity recognition. However, those systems still suffer from two limitations, preventing their real-world applications. First, due to the limitation of adversarial learning, existing systems require to identify different users (e.g., user ID/label) before they can perform a correct gesture recognition. Thus, these cross-target methods are often infeasible in practical use as the user ID is often unavailable to the system. Second, those learning-based systems require a large volume of training data to learn an efficient recognition model, but it is prohibitively time-consuming to collect a large amount of data due to user involvement. For example, it took 277 days to collect 1.2 million measurements for learning a cross-target model; even when each measurement only takes 20 seconds [49]. It is worth noting that, although there are some cross-environment gesture recognition systems [18, 45], those systems mainly rely on adversarial learning, and thus their solutions still have the limitations as we have discussed.

We present CrossGR, a deep-learning-based cross-target gesture recognition system for commercial Wi-Fi devices. CrossGR is designed to address the two aforementioned limitations of current cross-target recognition systems. To perform gesture recognition across different targets (i.e., users) without requiring having prior knowledge of them (i.e., knowing who is performing a gesture), CrossGR uses user-agnostic but gesture-related features from the wireless channel metric (CSI in this work). To that end, CrossGR employs a novel feature extraction framework by first employing a convolutional neural network (CNN) to extract all features from the CSI measurement and then using machine-learned classifiers to identify gesture-related features and remove the target-specific ones (e.g., human body and behavioral characteristics) from the CNN-extracted features. By doing so, we direct the learning framework to focus on using gesture-related, but user-agnostic signal representations to inform decision making. This in turn eliminates the impact of user domain features, making our gesture recognition system portable across different users.

While our new gesture-recognition framework provides a potentially powerful capability for learning useful signal representation, its potential can only be unlocked with sufficient training data. Typical deep learning algorithms require up to millions of examples to learn an efficient model [27], but collecting such a large number of training data through user involvement is often infeasible in practice. Our solution for avoiding expensive training data collection is to generate synthetic training data to complement those real-world samples collected from the end-users. To this end, we leverage the recent advances in the generative adversarial network (GAN) [15] that has shown impressive performance on image translation [17] and natural language [44] processing tasks. Our GAN-based training data synthesizer consists of two models: a generative model and a discriminative model. The generator is trained to capture and approximate the underlying distributions of real data collected from a few users for the targeting gestures. By sampling the distribution of the real data in a gesture-related feature space, we can produce many new synthetic training samples that follow the same distribution of the real-life samples. The discriminator is trained to estimate the probability that a sample comes from the real dataset but not the synthetic ones. By jointly optimizing the generator and the discriminator, the generator would become better in creating synthetic data that are similar to those seen in the real-world; and the discriminator would become better in recognizing synthetic data. At the end of this generation-discrimination training process, we will obtain a train data generator that is highly effective in generating synthetic training samples for the targeting features. We can then populate our training data with many training data, covering the space far more finely than what can be achieved by real data samples.

We have implemented a working prototype of CrossGR using commercial off-the-shelf Wi-Fi devices. We evaluate the performance of CrossGR in a typical classroom furnished with desks and chairs. We applied CrossGR to perform gesture recognition across ten users and 15 gestures. Our extensive experimental results show that CrossGR delivers an accuracy of over 82.6% (up to 99.75%). We show that such accuracy is comparable to the state-of-the-art gesture recognition system, but CrossGR achieves this by using an order of magnitude fewer training samples collected from the end-users.

Technical contributions. This paper makes the following contributions:

- It introduces a novel target-adaptive scheme to extract user-agnostic but gesture-related features from Wi-Fi channel measurement (Section 4.4). Our approach does not rely on prior knowledge (e.g., user ID/label) of the targets, significantly enhancing the practicability of the developed Wi-Fi sensing system.
- It is the first to employ a GAN-based approach to reduce the human involvement (and hence the associated cost) for collecting training samples to learn a gesture-recognition system (Section 4.3).
- It demonstrates how CrossGR can be implemented using commercial off-the-shelf Wi-Fi devices to deliver consistently good performance for gesture recognition (Section 7).

2 BACKGROUND

In this section, we introduce CSI and the GAN used in this work.

2.1 Channel Statement Information

In a wireless communication system, the CSI can be obtained from commodity Wi-Fi network interface cards (NICs) [42], e.g., Intel 5300, Atheros 9580, etc. CSI describes how a signal propagates from a transmitter to a receiver over multiple sub-carriers. Let $X(f,t)$ and $Y(f,t)$ represent the transmitted and received signals on the

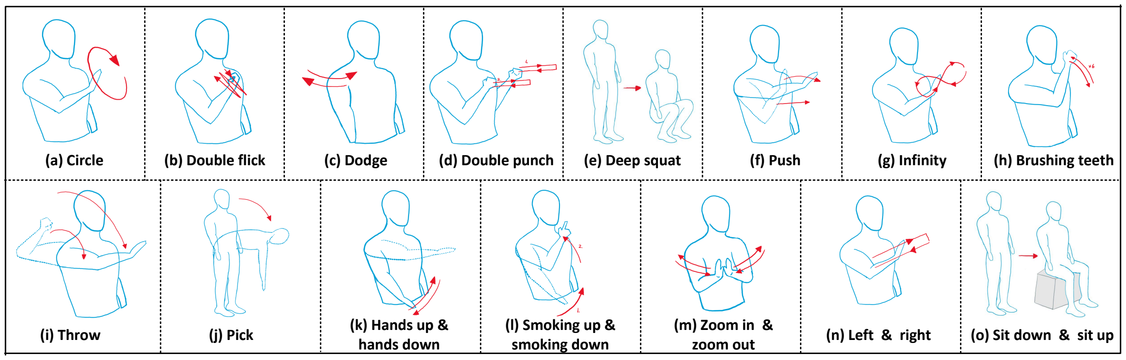


Fig. 1. Gestures considered in this work.

sub-carrier with frequency f at time t. Then, we have:

$$Y(f,t) = H(f,t) \times X(f,t), \tag{1}$$

where, the channel matrix $\mathbf{H} = (H(f_1,t),...,H(f_K,t))$ refers to the Channel State Information (CSI), and K is the number of sub-carriers. $H(f_k) = \|H(f_k)\|\, e^{j\theta_{f_k}}$ depicts the changes of amplitude $\|H(f_k)\|$ and phase θ_{f_k} via sub-carrier f_k. The Intel 5300 Wi-Fi card reports $K = 30$ OFDM sub-carriers for each transmission [11].

2.2 Generative Adversarial Network

Generative Adversarial Network (GAN) [15] is a neural network framework to generate new data that has the same statistics as the training data through an adversarial process. GAN consists of two fundamental models: a generative model G to capture the training data distribution, and a discriminative model D to estimate the probability of a sample that comes from the training data rather than G.

The network first learns the generator distribution P_G over training data $\mathbf{x}$. It defines a prior noise vector $\mathbf{z}$ with uniform or Gaussian distribution $P_{\mathbf{z}}(\mathbf{z})$, and builds a mapping function to the data space as $G(\mathbf{z};\theta_g)$. It also defines a discriminator $D(\mathbf{x};\theta_d)$ representing the probability that $\mathbf{x}$ comes from the training data rather than $P_G(\mathbf{x})$. The D and G are trained simultaneously by adjusting parameters for G to minimize $log(1 - D(G(\mathbf{z}))$, and for D to minimize $logD(\mathbf{x})$. Overall, the adversarial learning process is similar to a two-player minimax game [15] with the following objective function $V(G;D)$:

$$\min_G \max_D V\ (\text{D,G}) = \mathbb{E}_{\mathbf{x}\sim P_{Data}(\mathbf{x})}[\log D(\mathbf{x})] + \mathbb{E}_{\mathbf{z}\sim P_{\mathbf{z}}(\mathbf{z})}[\log(1 - D(G(\mathbf{z})))]. \tag{2}$$

3 MOTIVATION

In this section, we highlight two practical issues for building an accurate gesture recognition model: the lack of training data (Section 3.2) and the cost for labelling training data (Section 3.3).

3.1 Experimental Setup

In this work, we recruited 15 volunteers and collected the CSI measurements of 15 commonly seen gestures given in Fig. 1. The Wi-Fi transceivers are deployed 2 m apart and 1.5 m above the ground in a typical indoor environment. We repeat the data collection process 10 times for each gesture per participant.

In total, we obtained 2,250 CSI measurements (15 users ×15 gestures ×10 instances). We use 1,575 samples from ten volunteers as the training data and the rest 675 samples from 5 volunteers as the testing data.

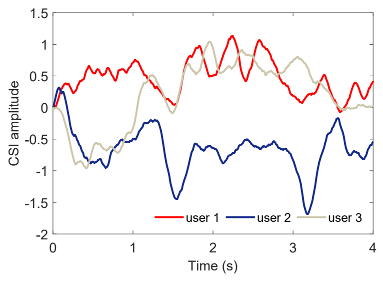


Fig. 2. The CSI patterns for different users.

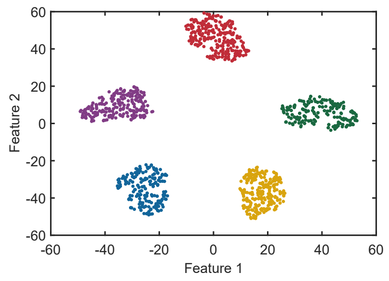


Fig. 3. Using t-SNE to cluster CSI data from different users.

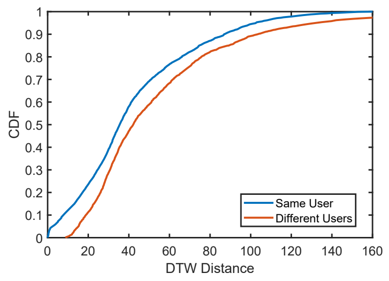


Fig. 4. DTW distance of Wi-Fi samples among the same user and different users.

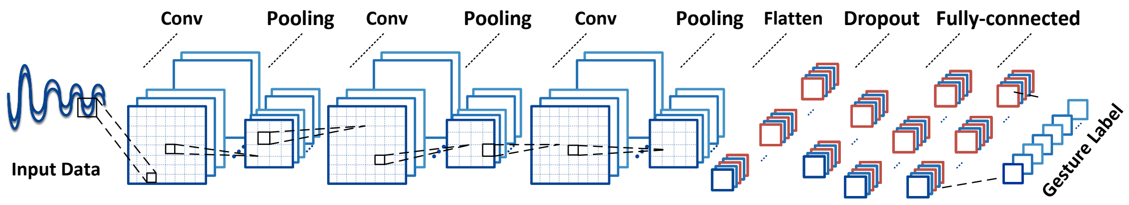


Fig. 5. Architecture of our CNN-based classifier.

3.2 The Issue of Lacking Training Samples

Fig. 2 shows the amplitude of CSI of Wi-Fi signals when three different users perform in the same gesture. As we can see, the three CSI amplitude sequences are visually different from each other for the same gesture. As a result, a model trained on data samples collected from a small number of users is unlikely to be effective for CSI signals collected from others because of the potentially drastic changes in the CSI signals. In an attempt to visualize this point, we apply t-SNE [22] to project the CSI data samples collected from five users onto a two-dimensional feature space, where samples belong to the same user share the same color code. As can be seen from Fig. 3, data samples form five distinct clusters according to from which user the data is obtained. Here, data samples are closer to each other from the same user on the feature space, but further away from those from different users. This diagram also suggests that a model learned on samples collected from a handful of users is unlikely to cover the problem space of many other users. The lack of neighboring observations means that the model is unable to reason about the behaviors of other users, which leads to data from other clusters to be incorrectly predicted. To quantify the differences among data samples, we compute the dynamic time warping (DTW) value of Wi-Fi signals collected between the same user and different users. The closer the DTW distance is, the more similar two measurements will be. The cumulative distribution function (CDF) diagram in Fig. 4 suggests that CSI measurements collected from the same user have a smaller DTW distance than those taken from different users. This reinforces our observation that using data samples collected from a few users is likely to lead to model overfitting due to the sparsity of training data.

To further elaborate on the impact of insufficient training data, we test the performance of a gesture recognition model in two scenarios. The first one is to train and test the model on data collected from the same target users. The second scenario is to train the model on data collected from some training users, but test the trained model on data of other users who were unseen during the training phase. We use 5-fold cross-validation to split the training and testing datasets. This means we partition the data into five sets. We use four sets for training the

Table 1. Signal features used in CrossGR.

Domain	**Features**
Time Domain	min, max, min/max 10th/90th, variance, mean, skewness, standard deviation, kurtosis, q-quantiles (q=0.25, 0.5, 0.75), inter-quartile range, etc. over a time window.
Frequence Domain	domain-frequency ratio, energy, FFT Peaks, etc.

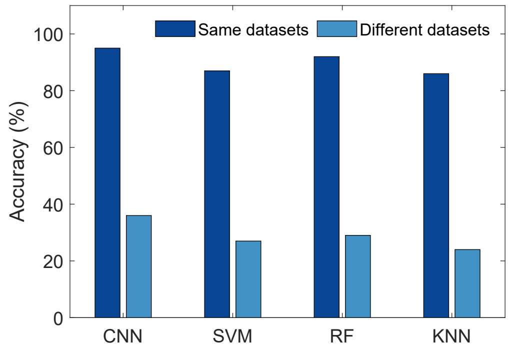


Fig. 6. Prediction accuracy with 5-fold cross-validation when the model training dataset includes samples from the test users (same datasets) and does not include samples from the test users (different datasets).

model and test the trained model on the remaining dataset. We repeat this process five times (folds) until each group of data is tested at least once and report the average accuracy across testing runs. For the second scenario, we partition the data by users.

In this experiment, we consider four widely used classifiers: a Convolutional Neural Network (CNN), a Support Vector Machine (SVM), K-Nearest Neighbors (KNN), and a Random Forest (RF) model. These models take as input features from both the time and the frequency domains given in Table 1. The detailed settings about the CNN classifier is given in Fig. 5. It can automatically extract informative features from the input CSI amplitude sequence. We use three convolutional layers and set the number of filters for each layer to 64. The size of the filters is 3, 5, 7, respectively. We use max pooling to down-sample the features in the pooling layer with a step size of 2. After flattening, two fully connected layers are used to extract features with ReLU as activation functions further. To reduce the overfitting, we add one dropout layer in which the rate is 0.5. Finally, the softmax Loss function as a classifier determines the label corresponding to the input.

Fig. 6 reports the average accuracy for each classifier under the two evaluation settings. As we can see from the diagram, all machine learning models perform poorly on unseen users. The CNN is best-performing model, but its prediction accuracy drops from 95% to 36% on previously unseen users. This is not supervising as machine learning models are brittle, and changes in the data distribution can lead to skewed prediction results. This is a problem known as out-of-distribution [16]. These results highlight the significant effect that the number and distribution of training programs have on the quality of predictive models. Without good coverage of the feature space, any machine learning methodology is unlikely to produce high-quality recognition models, leading to a poor generalization ability. To improve the generalization ability of a gesture recognition model, we need to find ways to generate sufficient training data to cover a complex, high dimensional problem. CrossGR is designed to offer such capability by generating synthetic training data.

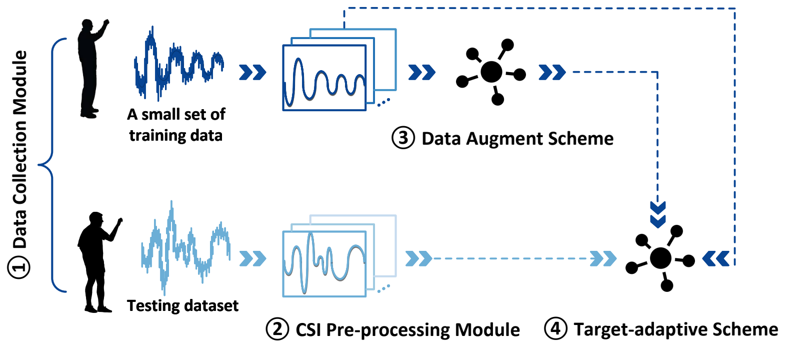


Fig. 7. An overview of CrossGR.

3.3 The Cost of Collecting Training Data

We have shown that a learned gesture recognition model could give poor performance when it is applied to new users. This is due to the limited number of training samples and the distribution of training data within the feature space. To improve the generalization ability of a model, we must provide sufficient training data to allow machine learning algorithms to model a complex, high-dimensional problem space. Unfortunately, collecting a large number of training samples involved a large number of users is extremely expensive. For example, the work presented in [49] had to collect 1.2 million activity samples to train an accurate model for 100 users. If we assume that collecting each sample takes 20 seconds, the whole process will take around 277 days - which is too expensive in practical use. Our novel approach, described in the next section, solves this problem by automatically generating an unbounded number of synthetic training samples. Our approach thus provides a low-cost method to generate training data to cover the feature space with fine granularity, which in turn allows one to build an accurate recognition model.

4 OUR APPROACH

This section presents the overview and the detailed design of CrossGR, including the CSI pre-processing module, data augment scheme, and target-adaptive scheme.

4.1 System Overview

CrossGR consists of the following four components, as depicted in Fig. 7.

- **Data Collection Module:** We first collect a small set of raw CSI measurements when different users move into the monitoring area. We then divide the collected measurements into the training dataset and testing dataset.
- **CSI Pre-processing Module:** After collecting data, we adopt the Butterworth filter to remove the high-frequency noise in the Wi-Fi signals. Then, we use the degree matrix and dynamic time warping (DTW) to select sub-carriers that are affected by the gesture. The detailed descriptions are in Section 4.2.
- **Data Augment Scheme:** To reduce labor costs, we apply the generative adversarial network (GAN) to the training data to generate a large number of synthesis signals that are similar to yet different from the collected ones. We also utilize a convolutional neural network to obtain fine-grained synthetic data. We detail the scheme in Section 4.3.

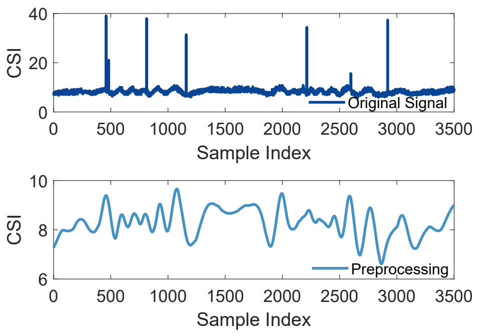


Fig. 8. Original and filtered CSI time series.

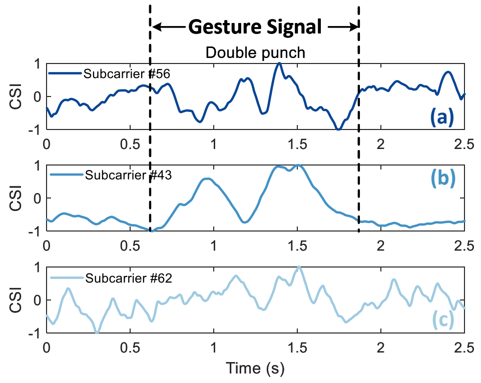


Fig. 9. The pattern of different sub-carriers.

- **Target-adaptive Scheme:** To avoid the requirement of prior-knowledge (i.e., user ID/label) for the targets, we design a target-adaptive scheme to remove target-related features and only keep gesture-related features. The detailed design is in Section 4.4.

4.2 CSI Pre-processing Module

CrossGR collects CSI measurements from the commodity Intel 5300 Wi-Fi card with 30 OFDM sub-carriers for each transmission. When the transmitter has 1 antenna and the receiver has 3 antennas, we get $1 \times 3 \times 30 = 90$ CSI time series.

4.2.1 Signal Denoise. The CSI values are noisy due to the defective hardware. It is crucial to remove the noise to achieve accurate gesture recognition. The key observation is that the noise has a higher frequency when compared to the gesture associated signal. In other words, the gesture induced frequency variations are at the low end of the spectrum. Accordingly, CrossGR employs the Butterworth filter on all sub-carriers in the time domain. In particular, with the carrier frequency of 5.825 GHz, the speed of a gesture with a frequency of 80 Hz is 2.06 m/s after considering the round-trip path length change. We set the cut-off frequency ω_s of the Butterworth filter as $\omega_s = 80/(f_n/2)$, where f_n is the sampling rate with the value of 1,000 in our experiments. Fig. 8 displays the amplitudes of the original CSI waveform and the resultant from the Butterworth filter. It shows that the Butterworth filter successfully removes most of the noise from the CSI measurements. After that, we normalize all the collected data.

4.2.2 Sub-carrier Selection. In practice, the Wi-Fi signals bounce off surrounding objects after being reflected from the human body, creating a multipath profile indicative of the human body and their activities. However, not all sub-carriers are affected equally by multipath reflections. We have two observations in the time domain:

- *Due to the frequency selective fading caused by multipath effects of wireless channels, the amplitude variations of different sub-carriers are inconsistent even for the same gesture.* For instance, as shown in Fig. 9(a) and Fig. 9(b), the 56th and 43rd CSI sub-carriers have completely different amplitude fluctuations between 0.6 s and 1.9 s in the "double punch" gesture. The 43rd sub-carrier showing two peaks is a more transparent illustration of how the gesture affects the wireless channel. Such a signal pattern provides a fine-grained basis for accurately recognizing gestures.
- *Due to the hardware noise, some sub-carriers are submerged and cannot reveal gesture information through amplitude changes in the time domain.* As exhibited in Fig. 9(c), the amplitude change of the 62nd CSI

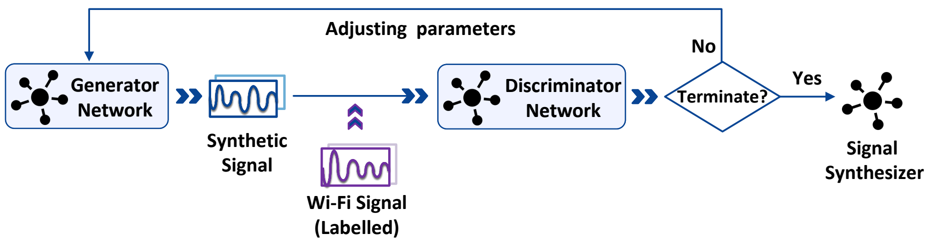


Fig. 10. The training process of our GAN-based wireless signal synthesizer.

sub-carrier is wholly immersed into the noise. Consequently, we cannot correctly identify the start point and endpoint of the "double punch" gesture.

The above observations illuminate that we can select sub-carriers to conduct accurate gesture recognition. Statistical analysis tells that most sub-carriers' amplitude variations efficiently reveal the influence of gestures on the wireless channel. Hence, to obtain more specific CSI sub-carriers, we utilize a degree matrix M_D to select signals and avoid introducing noise:

$$M_D^j = \sum_{j=1}^{n} W_{ij}, \tag{3}$$

where $W = \sum_{i=1}^{n} \sum_{j=1}^{n} DTW(i, j)$ is similarity matrix calculated using dynamic time warping (DTW) [32], n is the number of sub-carriers with a value of 90 in our experiment. Finally, we choose the first 30 sub-carriers given by degree matrix.

4.3 Data Augment Scheme

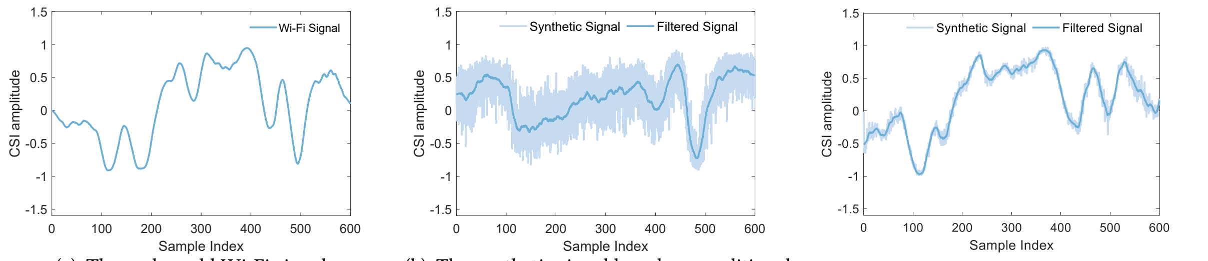
Usually, it requires a mass amount of CSI measurements to train an accurate gesture recognition model. However, the data collection process is time-consuming and labor-exhausting. Inspired by the GAN, CrossGR attempts to reduce human involvement and cost via a signal synthesizer. Fig. 10 illustrates the process of training a wireless signal synthesizer using GAN. The framework contains a generator G and a discriminator D. G is to synthesize pseudo data which has similar yet different distribution to the real training data. D is to verify that the data sample is from the real data rather than the generator. The optimization between the generator and the discriminator ensures that the generated data has the most approximate distribution to the training data.

Specifically, the generator and discriminator work similarly to that conditional GAN [24]. Based on Equation 2, the objective function becomes the following:

$$\min_G \max_D V(D, G) = \mathbb{E}_{\mathbf{x} \sim P_{Data}(\mathbf{x})} [\log D(\mathbf{x}|\mathbf{y})] + \mathbb{E}_{\mathbf{z} \sim P_{\mathbf{z}}(\mathbf{z})} [\log(1 - D(G(\mathbf{z}|\mathbf{y})))] . \tag{4}$$

4.3.1 Data Formulation. We provide a general description of the data definition. Given a training dataset $X^r = \{x^{r_1}, x^{r_2}, ..., x^{r_M}\}$ with the corresponding label $Y = \left\{y^1, y^2, ...y^K\right\}$, K is the number of gesture categories, and a testing dataset $X^u = \{x^{u_1}, x^{u_2}, ..., x^{u_N}\}$ without identity labels. The generator synthesizes signals based on a label from Y as $X^g = \{x^{g_1}, x^{g_2}, ..., x^{g_L}\}$, where the value of M, N and L may not equal. For simplicity in description, we define an overall dataset X, which is the union of X^rn X^u and X^g, i.e.,

$$X = X^r \sqcup X^u \sqcup X^g. \tag{5}$$



(a) The real-world Wi-Fi signal.

(b) The synthetic signal based on conditional GAN.

(c) The synthetic signal after convolution operations.

Fig. 11. Illustrations of the real-world Wi-Fi signal and the synthetic signal.

4.3.2 Signal Generator. The generator G automatically generates outputs mapped from a random noise vector z^g with label y. The goal is to synthesize the pseudo data x^g which can not be distinguished from real data distributions, $G : (z^g, y) \longrightarrow x^g$. However, unlike the precise synthetic image and natural language by GAN, the synthetic wireless signal is more easily distorted by noise during the generation process. We take one sub-carrier as an example. Compared with original Wi-Fi signals in Fig. 11(a), the synthetic data is noisy in Fig. 11(b). Noise components retain even if the synthetic signal is refined using a Savitzky-Golay filter (also known as least-square smoothing filter). What's worse, the extracted feature from synthetic data is not only bound to gesture information but also include noise information brought by the process of fitting distribution.

To make the generator better fit the original data distribution, we add convolution operations to extract efficient features [31] automatically. The convolution operation can make the synthetic data more distinct while restoring the specific characteristics, thus resolving noise interference. Also, we utilize the Savitzky-Golay filter to remove the residual noise further. Fig. 11(c) indicates that the synthetic data is more fine-grained than the output of GAN in Fig. 11(b) and the filtered data becomes smoother. Note that synthetic data is labelled. We update generative network parameters with the following loss function:

$$L_g = -\frac{1}{m} \sum_{(z^g, y)} \log(D(G(z^g, y))), \tag{6}$$

where m is the number of noise samples in a minibatch.

4.3.3 Signal Discriminator. The discriminator D takes $< x^r, y >$ and $< x^g, y >$ as inputs and gives the probability of the input being synthetic data. Like the generator, to make the discriminator better to extract useful features, we introduce convolution operations to improve discrimination accuracy. In order to reach a better antagonistic result, we use batched signals to train the discriminator to identify real data and synthetic data. We update the parameters with the following loss function:

$$\mathrm{L}_{d_g} = -\frac{1}{m} \left[\sum_{x^g, y} \log\left(1 - D\left(x^g, y\right)\right) + \sum_{x^r, y} \log D\left(x^r, y\right) \right], \tag{7}$$

where $D(x, y) = R/F$, is the probability of the input being a synthetic Wi-Fi signal, and $1 - D(x, y)$ is that of a real one. R and F means from the real data and synthetic data, respectively.

4.3.4 Training the Network. In the training stage, we iteratively update the parameters until taking the minimum overall loss function. We employ an alternate method to train G and D. It also has a two-player minimax game, and the generation-discrimination process terminates when the discriminator can not distinguish the synthetic

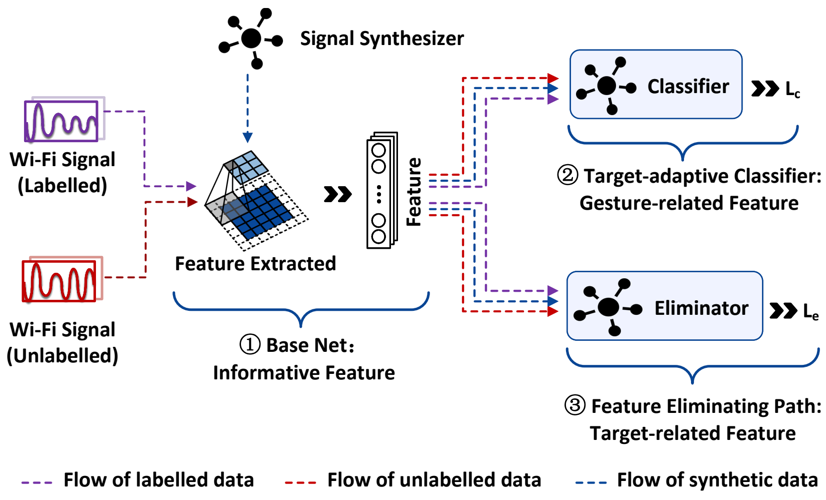


Fig. 12. Overview of the target-adaptive scheme.

signals from the real ones. Note that we select Stochastic Gradient Descent(SGD) with a learning rate of 0.0002 to optimize network parameters.

4.4 Target-adaptive Scheme

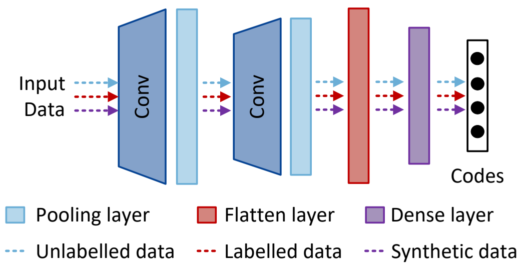
The raw CSI measurements contain not only signal channel information resulting from gesture activities, but also multi-paths resulting from specific user characteristics (e.g., the height, weight, and body shape of the target user) and the environment. We would like to filter out the user and environmental characteristics from the CSI measurements because we want to make the decision-making process of our gesture recognition model to be independent of the user and the environment. To this end, we design a target-adaptive scheme that automatically removes the target-related features and only keeps gesture-related features. This process is illustrated in Fig. 12. It includes three parts: two branches and one base network. The base network first uses a stack of shared convolutional layers to extract all features related to both targets and gestures from CSI measurements. Then, one branch named "target-adaptive classifier" aims to identify and extract gesture-related features, and the other branch called "feature estimating path" seeks to remove the noisy target-related features.

4.4.1 Base Network. A convolutional neural network can extract useful features from wireless signals thanks to its superior feature learning capabilities [45, 46]. Therefore, we use a stack of shared convolutional layers as the feature extraction layers in base network, which aims to learn general but informative representations of Wi-Fi signals before target-adaptive classifier and feature eliminating path, as shown in Fig. 13(a). Given the input signals X, the feature representation F is as follow:

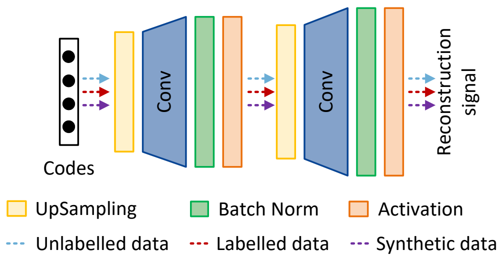
$$\mathrm{F_b} = \mathrm{CNN}\left(X; \theta_b\right), \tag{8}$$

where θ_b is the set of all parameters in CNNs.

However, it is challenging to extract general but informative representations for labelled and unlabelled data. To do so, we reconstruct the extracted features into signals based on a reconstruction path after the feature extraction layers. The reconstructed signals try to reproduce the original data as much as possible. As shown in Fig. 13(b), we first upsample the input. Then we use 2D kernels as filters at each convolutional layer for each CNN block and the batch norm layer to normalize each layer. At last, an activation function ReLU is to introduce



(a) Feature Extraction Layers.



(b) Reconstruction Layers.

Fig. 13. Structure of base network.

the non-linearity relationship between the layers of the neural network. Previous approaches show that a more traditional loss, such as $L1$ or $L2$ distance, can improve performance. However, it only considers the difference between the reconstructed signal point and the sample point, neglecting the relationship between those points in a small area. As a result, the output of the reconstruction layers is overconfidence and unsmooth. Considering the continuity of each sub-carrier and the correlation between sub-carriers, and inspired by computer vision, we propose a point-correction loss function to improve the model's performance:

$$L_{pair} = \frac{1}{n}\sum_{i=1}^{n}\left(\log x_i^* - \log x_i\right)^2 - \frac{1}{n^2}\left(\sum_{i=1}^{n}\left(\log x_i^* - \log x_i\right)\right)^2, \tag{9}$$

where x^* is the reconstructed signal, and x is the ground truth. This function not only focuses on the relationship between the reconstructed signal points and the sample points but also the correlation among these points in a small area. It will automatically correct points that are not predicted rightly.

4.4.2 Target-adaptive Classifier. After the base network, we obtain all features consisting of the gesture-related features and the target-related features. We use a target-adaptive classifier to identify the gesture label to obtain the common gesture-related features across different users. As we have described earlier, our key insight is that the higher the classifier accuracy is, the more precise the features are in capturing the gesture-specific characteristics. As a result, training a highly accurate classifier also helps us in extracting fine-grained, gesture-related features. As exhibited in Fig. 14, based on the outputs of the base network (i.e., F_b), we utilize two convolution layers with 2D kernel and a fully connected layer followed by an activation function to extract the representation further F_c:

$$\mathrm{F_c} = \mathrm{CNN}\left(F_b; \theta_c\right), \tag{10}$$

where θ_c is the set of all parameters. To predict the gesture label, we use the softmax function to non-linearly map F_c to the K-dimensional gesture prediction distribution, which corresponds to the K labels provided by the training dataset in our system:

$$y_c = softmax(W_c F_c + b_c), \tag{11}$$

where W_c and b_c are parameters. The labelled and unlabelled data are fed into the target-adaptive classifier, thus $y_c = \left[y_c^l, y_c^u\right]$, where y_c^l represents the predicted probabilities of labelled data (i.e. real signals and synthetic signals) and y_c^u represents the predicted probabilities of unlabelled data (i.e. test signals). For labelled data, the predicted distribution y_c^l is compared to the ground truth y via the cross-entropy loss as follow:

$$\mathrm{L}_c = \sum_{i=1}^{K} -y^i \log y_c^{l_i}. \tag{12}$$

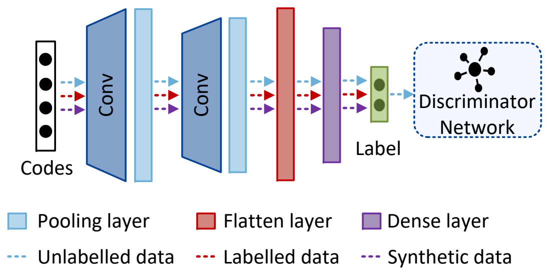


Fig. 14. Structure of target-adaptive classifier.

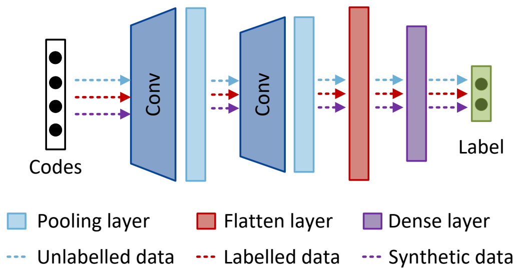


Fig. 15. Structure of feature eliminating branch.

Unlike labelled data, the cross-entropy for unlabeled data cannot be calculated because the explicit label is unknown in advance. Nevertheless, inspired by adversarial learning, we try to put them into the discriminator of the GAN. We attempt to use the discriminator to identify the authenticity of unlabeled data and the predicted label. The discriminator identifies the synthetic data with the real labels (i.e., $< x^g, y >$) from the generator and the real data with the predicted labels (i.e., $< x^u, y_c^u >$) from the classifier as false. Note that the unlabelled data is the real data collected from the real-world. The classifier aims to predict a pseudo label given the real data. Therefore, we can verify the output of classifier through the discriminator:

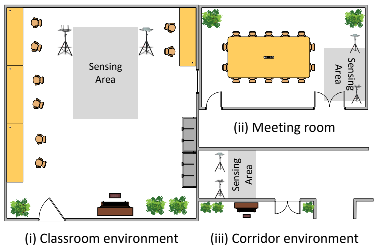
$$\mathrm{L}_{d_c} = -\frac{1}{m} \sum_{x^u, y^c} \log \left(1 - D\left(x^u, y^c\right)\right), \tag{13}$$

and thus we update the L_{d_c} with the regularization term λ_d set to 0.0001 as follow:

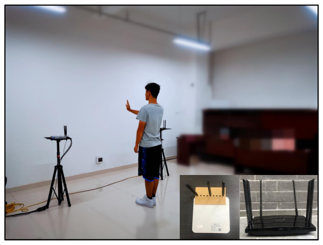
$$\mathrm{L}_d = \mathrm{L}_{d_g} + \lambda_d \mathrm{L}_{d_c}. \tag{14}$$

4.4.3 Feature Eliminating Path. As depicted in Fig. 12, we use an auxiliary network, namely the feature eliminating path, to guide the base network to focus on extracting gesture-related features to feed into the target-adaptive classifier for downstream gesture recognition. Note that this auxiliary network is *merely* used during the training phase to assist feature extraction. It does not participate in gesture classification once the entire network is trained and deployed. Our goal here is to find a set of features that can maximize the discriminative information across gesture classes. Our key insight is that that user- and environment-specific characteristics are likely to be gesture-agnostic. As a result, they would be relatively consistent regardless of what gesture is being performed. If we could identify such characteristics, we could then be able to remove them from the feature set given by the base network.

To that end, our feature eliminating path network aims to quantify if we feed a specific set of features (given by the base network), how will that affect the probability distribution given by a softmax layer of the feature eliminating path network. This is done by measuring the difference in the probability distribution across the gesture labels using a dedicated loss function. When considering N (e.g., 15 in this work) gestures, the softmax layer of the feature eliminating path component will output a probability distribution across N classes, with all the probabilities sum up to 1.0. If the features fed into the softmax layer are largely gesture-independent, we would expect all gesture classes to have more or less the same probability (i.e., close to $1.0/N$). If this is the case, it means the features used for classification do not provide sufficiently discriminating information to distinguish different gesture classes; and hence should not be used for gesture prediction. By back-propagating such findings to the base network during the training phase, we can guide the base network to extract feature-relevant information and ignore that is not.



(a) Layout of evaluation environments.



(b) Example of experimental setting.

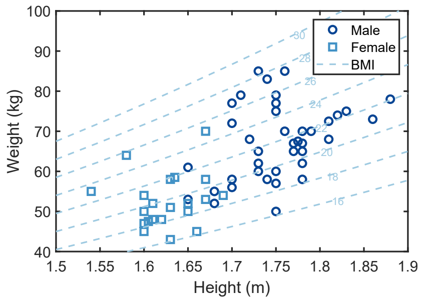


Fig. 16. The experimental setup layout in (a) and experimental scenario in (b).

Fig. 17. The somatotype information of our participants.

Our feature eliminating path is a subnet of our neural network, as shown in Fig. 15. It consists of two convolution layers and a fully connected layer. The representation F_e after the subnet is as follow:

$$\mathrm{F}_e\!= \mathrm{CNN}\left(F_b; \theta_e\right), \tag{15}$$

where θ_e is the set of all parameters. After that, we also use the softmax function to map F_e to the gesture prediction distribution non-linearly:

$$y_e = softmax(W_e F_e + b_e), \tag{16}$$

where W_e and b_e are parameters. To enable accurate extraction of target-related features, we reject the precise classification probability in a specific category. Therefore, we compare the predicted distribution y_e with the ground truth, which equals to $\frac{1}{K}$ via the cross-entropy loss as follow:

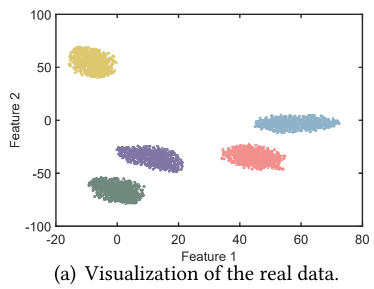
$$\mathrm{L}_e = -\frac{1}{K}\sum_{i=1}^{K} \log y_e^i. \tag{17}$$

In essence, loss function L_e extracts gesture-agnostic features by trying to assign equal probabilities to different classes of CSI measurements. Note that we take both labelled (i.e., real and synthetic signals) and unlabelled data as inputs. In this way, after the unlabelled data tune the parameters of the feature eliminating path and base network, the target-adaptive classifier is forced to predict labels with features representing unlabelled distributions.

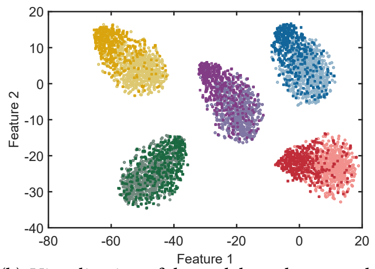
5 IMPLEMENTATION

Hardware setup. We build a prototype of CrossGR using one transmitter and one receiver. The transmitter is a TP-link TL-WR890N 450M router with three antennas work in 802.11n AP mode at a 5.825 GHz frequency band. The receiver is a laptop with an Intel 5300 Wi-Fi NIC and one antenna. The laptop is configured with a 3.6 GHz CPU (Intel i7-4790) and an 8 GB memory. The transmission rate in our experiment is set as 1,000 packets per second which are widely used in past gesture recognition systems [30, 49]. Our software prototype is implemented as Matlab and Python programs.

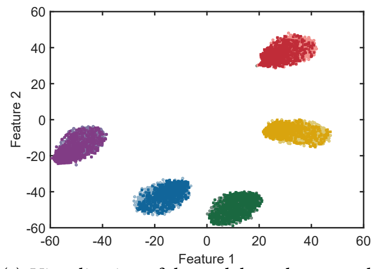
Environment setup. We conducted extensive gesture recognition experiments in three indoor environments to evaluate the performance of CrossGR: a classroom (default setup) furnished with desks and chairs, a spacious corridor environment and a meeting room with furniture like tables and chairs. Fig. 16 illustrates the layout of those environments with a 2 m × 2 m sensing area. The laptop is placed 2 m away from the router and all devices are held up at a height of 1.2 m.



(a) Visualization of the real data.



(b) Visualization of the real data plus a number of 7, 500 synthetic data.



(c) Visualization of the real data plus a number of 15, 000 synthetic data.

Fig. 18. t-SNE visualization of the real-world data and the synthetic data. Different symbol colors denote different gestures. Specifically, blue stands for circle, purple stands for dodge, green stands for the double flick, yellow stands for double punch, red stands for hands up and hands down.

Dataset. We invited 60 volunteers to participate in our experiment. The participants have different heights, weights, and somatotype, as shown in Fig. 17. Specifically, there are 39 males whose height varies from 1.65 m to 1.88 m and weight varies from 50 kg to 85 kg; 21 females whose height varied from 1.54 m to 1.69 m and weight varies from 43 kg to 70 kg. We consider a total of 15 gestures as shown earlier in Fig. 1. To collect CSI data for gesture recognition, each participant performs each gesture 20 times. We obtain 18, 000 CSI measurements in total (60 users $\times$ 15 gestures $\times$ 10 instances). In the default experimental setting, we split 16, 200 samples of fifty-four volunteers as the training data and the rest 1, 800 samples of six volunteers as the testing data. Moreover, we use GAN to generate 30, 000 synthetic samples for each gesture to expand training data.

6 MICRO-BENCHMARK

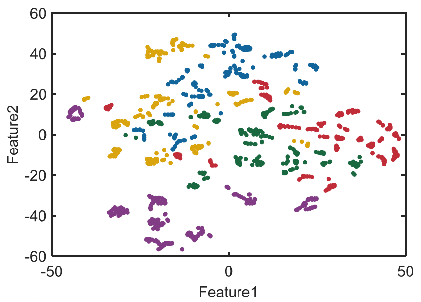
To illustrate the effectiveness of the data augment scheme and target-adaptive scheme, we run the following two benchmark experiments.

6.1 Verification of Data Augment Scheme

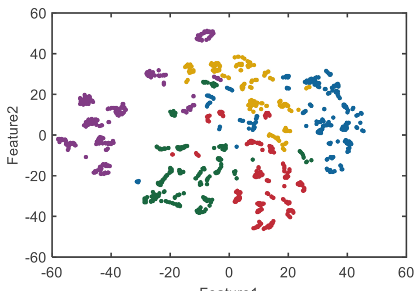
In order to verify the effectiveness of GAN in generating data, we use t-SNE [22] to conduct joint dimension reduction processing for the original Wi-Fi signal and the synthetic data generated. We choose five gestures for illustrations. Firstly, we reduce the dimensionality of collected Wi-Fi signals from the real world, as shown in Fig. 18(a). Ideally, those synthetic data should be as close to their corresponding real data as possible. Therefore, we perform joint dimensionality reduction on synthetic data and real data. Fig. 18(b) and Fig. 18(c) demonstrate that, for each class, their real data and synthetic data are well-aligned. It implies that GAN can fit the original Wi-Fi signal distribution well and expand the diversity of data to a certain extent.

6.2 Verification of Target-Adaptive Scheme

To verify the effectiveness of our target-adaptive scheme, we compare our scheme against the well-known CNN method. The framework of a CNN-based neural network is shown in Section 3. In this benchmark, we take 5 gestures as an example. Ideally, there should be five aggregated categories, i.e., each category shows the features of one gesture. We first visualize and compare features extracted by the existing CNN method and our target-adaptive scheme via t-SNE. Fig. 19(a) shows the features extracted by the CNN method. As we can see, different categories are overlapped because these features contain both target-related features and gesture-related features. Fig. 19(b) shows features extracted by our target-adaptive scheme. Now, there are obvious boundaries between



(a) Visualization of CNN extracted features.



(b) Visualization of our target-adaptive scheme extracted features.

	A	B	C	D	K
A	47%		18%	16%	19%
B	27%	48%	17%	4%	4%
C	30%		51%	15%	4%
D	26%	16%	22%	32%	4%
K	6%	3%	9%	5%	77%

(c) The accuracy based on CNN extracted features.

	A	B	C	D	K
A	92%		4%	1%	2%
B	3%	93%	4%		
C	3%	3%	90%	4%	
D	6%	3%	3%	88%	2%
K	1%	1%	4%	3%	93%

(d) The accuracy based on our scheme extracted features.

Fig. 19. Comparisons between our scheme and CNN in feature extraction and recognition across 5 different targets. There are five gestures: blue stands for circle, red stands for dodge, green stands for the double flick, yellow stands for double punch, purple stands for hands up and hands down.

different categories because our scheme can remove the target-related features and only keep the gesture-related features. Then, we compare the recognition accuracies when using features extracted by CNN and our scheme. Fig. 19(c) shows that the accuracy of CNN extracted features is as low as 32% for identifying five gestures. In contrast, the accuracy of our scheme extracted features is more than 88%, as shown in Fig. 19(d). Overall, the two experiments demonstrate the advantages of our target-adaptive scheme in removing target-related features and achieving a high accuracy across different targets.

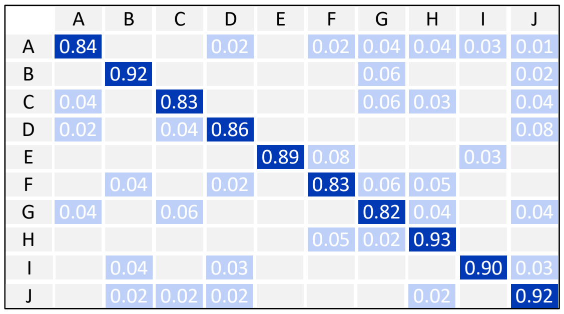
7 PERFORMANCE EVALUATION

7.1 Overall Performance

To evaluate the overall performance of CrossGR, we conduct two experiments when the number of gestures is 10 and 15, respectively. For the two experiments, we have six different targets/users and use the same default experimental setting. Fig. 20 shows confusion matrices of the recognition accuracy for different gestures. The average accuracy of our system is 87.4% and 82.6% when the number of gestures is 10 and 15. It implies that our system can achieve high accuracy for cross-target gesture recognition.

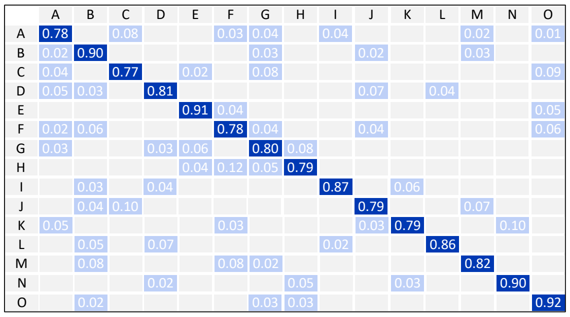
7.2 Comparisons with Existing Recognition Models

To illustrate the advantages of CrossGR, we compare CrossGR against several alternative recognition approaches, namely Support Vector Machine (SVM), K-Nearest Neighbors (KNN), Random Forest (RF), and Convolutional Neural Network (CNN). The first three approaches (i.e., SVM, KNN, and RF) use traditional features (listed in Table 1) for gesture recognition. The CNN and CrossGR extract deep features for gesture recognition. In the



	A	B	C	D	E	F	G	H	I	J
A	0.84			0.02		0.02	0.04	0.04	0.03	0.01
B		0.92					0.06			0.02
C	0.04		0.83				0.06	0.03		0.04
D	0.02		0.04	0.86						0.08
E					0.89	0.08			0.03	
F		0.04		0.02		0.83	0.06	0.05		
G	0.04		0.06				0.82	0.04		0.04
H						0.05	0.02	0.93		
I		0.04		0.03					0.90	0.03
J		0.02	0.02	0.02				0.02		0.92

(a) The confusion matrix with 10 gesture datasets.



	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
A	0.78		0.08			0.03	0.04		0.04				0.02		0.01
B	0.02	0.90					0.03			0.02			0.03		
C	0.04		0.77		0.02		0.08								0.09
D	0.05	0.03		0.81						0.07		0.04			
E					0.91	0.04									0.05
F	0.02	0.06				0.78	0.04			0.04					0.06
G	0.03			0.03	0.06		0.80	0.08							
H					0.04	0.12	0.05	0.79							
I		0.03		0.04					0.87		0.06				
J		0.04	0.10							0.79			0.07		
K	0.05					0.03				0.03	0.79			0.10	
L		0.05		0.07					0.02			0.86			
M		0.08				0.08	0.02						0.82		
N				0.02				0.05			0.03			0.90	
O		0.02					0.03	0.03							0.92

(b) The confusion matrix with 15 gesture datasets.

Fig. 20. The confusion matrix of gesture recognition accuracy across six different targets/users.

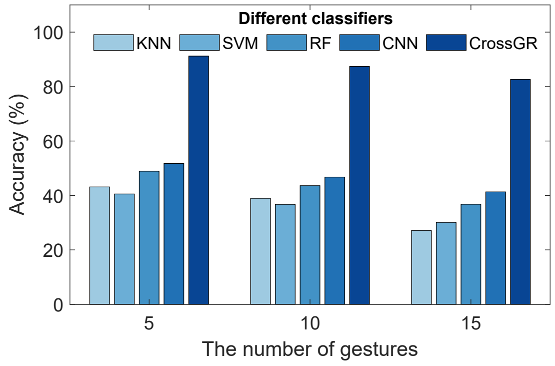


Fig. 21. Comparisons with existing recognition approaches.

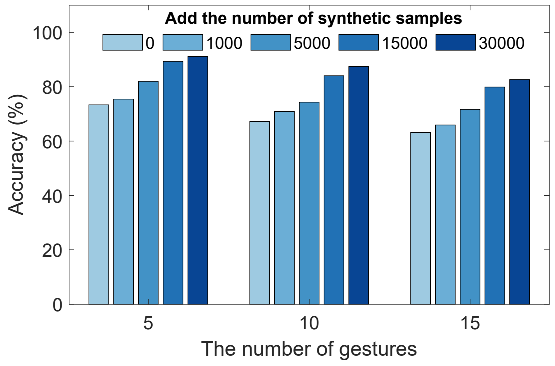


Fig. 22. Impact of the number of synthetic data samples.

experiment, we use the default experimental setting for training and testing datasets. Fig. 21 shows the gesture recognition accuracy of the five approaches. All existing approaches output disappointing results, i.e., less than 50% accuracy for most cases. Because existing approaches can not remove the features that are related to the human body and behavioral characteristics, resulting in large errors for new users. In contrast, CrossGR achieves the best accuracy by extracting unique gesture-related features and removing the noisy target-related features.

7.3 Impact of the Amount of Synthetic Data

In this section, we evaluate the impact of the amount of synthetic data added for training our system. To do so, we evaluate the accuracy of our system when the number of synthetic data is 0, 1, 000, 5, 000, 15, 000, and 30, 000. For each number of synthetic data, our system is tested with a number of 5 gestures, 10 gestures, and 15 gestures. The results are shown in Fig. 22. As we can see, when we only use the collected data to train the model, i.e., the number of synthetic data added is 0, CrossGR only achieves an average accuracy of 73% for 5 gestures. While the average accuracy increased to 91% when the number of synthetic data grows to 30, 000. When the number of gestures is 10 and 15, we also see a similar accuracy increase trend by adding more synthetic data. It demonstrates that the generated synthetic data can effectively improve the system accuracy.

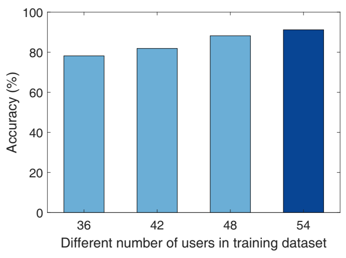


Fig. 23. Impact of the number of users in training datasets.

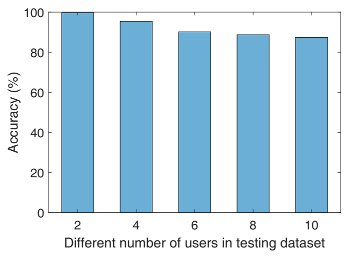


Fig. 24. Impact of the number of users in testing datasets.

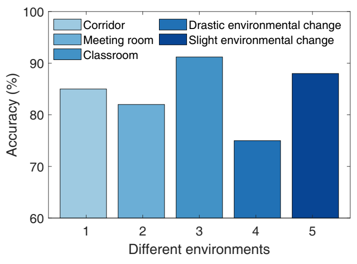


Fig. 25. The performance of different environments.

7.4 Impact of Training Dataset Diversity

Signals collected from the real world are the library for training GAN and generating synthetic data. Therefore, different amounts of real data have different effects on the performance of the system. We conduct gesture recognition experiments by varying the number of training datasets, and evaluating the impact of the number of datasets on the system performance. Specifically, we changed the number of users in the training dataset from 36 to 54. Each user performs each gesture 20 times. We take 5 gestures as an example and use GAN to generate 30, 000 synthetic samples as the training data in all experiments. The number of users in the testing dataset is fixed as 6. Fig. 23 shows that the accuracy increases when the number of users increases in the training dataset. For example, the accuracy increases by nearly 14% when the number of users increases from 36 to 48.

7.5 Impact of Testing Dataset Diversity

This experiment studies the impact of the number of volunteers in the testing dataset on the system performance. We take five gestures as an example and vary the number of volunteers from 2 to 10 in the testing dataset. We use CSI measurements collected from 50 volunteers and 30, 000 synthetic samples for each gesture category as training data. Fig. 24 shows that the system achieves the best accuracy of 99.75% when there are only two test users and relatively high accuracy of 87.42% when the number of testing users increases to 10. Overall, these results demonstrate that CrossGR is robust to the growth of testing users number.

7.6 Evaluation in Different Environments

In this experiment, we applied CrossGR to 5 gestures in three environmental settings: a classroom, a meeting room, and a corridor environment, as shown in Fig 16. We collect CSI measurements of gesture activities performed by the same six participants in the three environments. We train CrossGR and test CrossGR using data collected from each environment. Fig. 25 reports the performance of CrossGR in different environments. CrossGR gives consistent performance across different environments, giving over 80% of the prediction accuracy. This evaluation suggests our approach is generally applicable, giving consistent good performance across different environments.

To evaluate the impact of a changing environment, we moved furniture in both the classroom (by moving the chairs at least 1 meter away from their original position) and the meeting room (by moving the chairs into the sensing area). We tested how the change of environment affected a deployed model. We report the performance of both settings in in Fig. 25, where we denote the classroom experiment as "slight environmental change" and the meeting experiment as "drastic environmental change". As expected, the environmental changes have an impact on CrossGR 's performance. For the classroom experiment, we observe a modest decrease in recognition accuracy, less than 4%. For the meeting room experiment, we see a slightly higher drop in the accuracy of 7%, because the chairs were moved into the sensing area. Since our work focuses on cross-target prediction, CrossGR

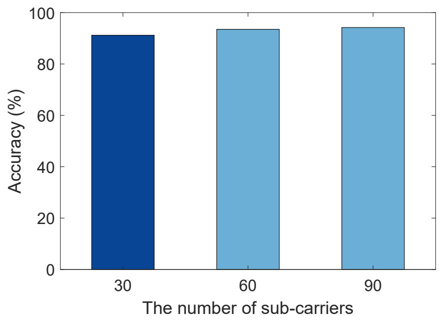


Fig. 26. Impact of the number of sub-carriers.

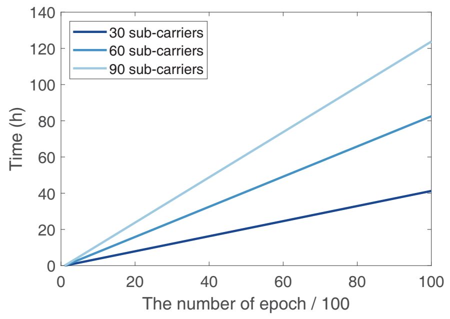


Fig. 27. Model training time with different numbers of sub-carriers used.

is not designed to address drastic environmental changes and has a limitation in adapting to environmental changes. Nonetheless, the overall performance of the system is still robust when the environmental changed is small.

7.7 Impact of the Number of Sub-carriers

We now evaluate how the number of sub-carriers affects the accuracy and training overhead of CrossGR. To do so, we varied the number of sub-carriers from 30 to 90. Fig. 26 and Fig. 27 respectively show the impact of the sub-carrier number on the model accuracy and training time. We can observe that using 30 sub-carriers is a sweet spot between training overhead and model accuracy. Using 90 sub-carriers incurs three folds more training overhead, but only yields a 3% improvement in the prediction accuracy. Using a higher number of sub-carriers as the model inputs also require more training data and training epoch. We choose to use 30 sub-carriers as our model inputs because it gives a good trade-off between model accuracy and training overhead.

8 DISCUSSIONS

CrossGR is among the first attempts to tackle cross-target gesture recognition with a limited number of real training examples. Naturally, there is room for improvement and future work. We discuss a few points here.

Mode collapse. GANs can suffer from a common problem called *mode collapse* where the generator may learn to produce some specific types of outputs that seem most plausible to the discriminator [23]. This is typically due to a local minimum in the discriminator learning process. This problem is just as true for our GAN model. As we have seen in Fig. 22, the accuracy of CrossGR reaches a plateau when using $15,000$ training samples. Using more training samples beyond this point yields little performance benefit (less than 2% improvement in accuracy). This is because the generator starts producing a similar set of synthetic data samples beyond $15,000$, and the additional samples offer little performance gain for the downstream machine learning model. There are numerous strategies to remedy this issue [25]. Adopting these methods to improve the quality of the synthetic data generated by CrossGR is our future work.

Interpretability. Machine learning techniques, in general, have the problem of relying on black boxes. Theoretical analysis for the capability and boundaries of a deep neural network is currently an active research field [5, 43]. Providing a theoretical proof of the underlying working mechanism of CrossGR is beyond the scope of this work, and is our future work. One way to gain insight into why the model fails to produce the desired result is to train an interpretable model (or the so-called surrogate models) like linear regressor to approximate the predictions of the underlying black-box model [6].

Environmental robustness. CrossGR has been evaluated in relatively static environments with no interference from other people around, where the environment changes little over time. How to improve the robustness of wireless sensing in a drastically changing environment is a currently active research field [51]. We will look into integrating some of the recent advances in this line of research to CrossGR and evaluate the performance of the resulting system in a more complex and dynamic environment.

9 RELATED WORK

Gesture and activity recognition is a typical application in wireless sensing. Over the past decades, many researchers have proposed several systems for recognizing human gestures and activities. They can be roughly categorized into wearable sensor-based, vision-based, and wireless-based.

Wearable sensor based. Current wearable device-based sensing systems detect human activities through various sensing devices [13, 14, 36]. For example, HeadScan [14] uses wearable sensors to capture and recognize human activities. BodyScan [13] can recognize many activities when a user carries a smartphone and wears a wristband/smartwatch. However, all these methods require on-body sensors.

Vision based. Vision-based gesture recognition systems capture images from cameras to track human motion and have been widely studied due to their high accuracy and universality [1, 37, 50]. However, they all have the following defects. Firstly, those systems are limited to the light condition and the viewing angle in the environment. Secondly, they can not work appropriately in non-line-of-sight situations. Thirdly, there are security issues such as privacy leaks in practical applications.

Wireless based. Recent years have witnessed many gesture and activity recognition systems based on the wireless signal. These systems don't require targets to wear any device, are not affected by light conditions, have less privacy leakage, and can capture the fine-grained changes in wireless channels [2, 4, 8, 20, 21, 26, 28, 29, 39–41, 48]. E-eyes [41] identifies the activity by comparing the CSI measurements against known profiles. CARM [40] quantifies the correlation between CSI dynamics and human activities. Zhang et al. [48] propose a diffraction-based sensing model to establish a relationship between CSI readings and human activity in the First Fresnel Zone(FFZ). WiMU [34] recognizes simultaneously performed gestures on commodity Wi-Fi devices. Unfortunately, none of these systems can cross-target and cross-environment because the features used for identification depend on human characteristics and environment.

Many innovations have been developed to adapt the recognition system in various domains [9, 33]. WiAG [35] lies in the translation function to generate virtual training samples for different locations. Widar3.0 [51] uses a model to recognize gestures since they design a novel environment-independent feature named body-coordinate velocity profile (BVP) that can generalize the ability of cross-domain. CrossSense [49] uses a machine learning model to generate training samples for cross-site sensing, and EI [18] removes environmental factors from CSI by using the adversarial network. Unfortunately, it is impractical to transfer these approaches to overcome cross-target issues directly. They either need to re-train the model for a new domain or restrict the user's behavior in advance when performing gestures.

Recently, some state-of-the-art works focus on the cross-target issue. For example, EUIGR [45] suppresses environment-related factors and obtains the user-specific features via adversarial learning but requires the user ID information in advance. FiDo [10] uses a joint classification-reconstruction structure to predict locations for new users and uses Variational Autoencoders (VAEs) to generate synthetic fingerprints. Although it does not require a label of the user domain, it cannot reconstruct the unique gesture-related features, resulting in the unstable performance.

These systems achieve expected performance on cross-domain. However, they fail to take into account the challenge of cross-target, and they all require intensive data collection. As a departure from the prior work, CrossGR works effectively for overcoming dataset bias and employs a GAN to generate an unbound number of data from a small set of Wi-Fi training measurements.

10 CONCLUSION

We have presented CrossGR, a novel cross-target Wi-Fi gesture recognition system. CrossGR aims to significantly reduce the overhead of collecting gesture training data across users. It is designed to work with no prior knowledge (such as who is performing gestures) of the target. It achieves this by using the deep neural network to extract the user-agonistic gesture features from the Wi-Fi channel information. To provide sufficient data to train the gesture-recognition model, CrossGR employs a generative adversarial network to first learn the distribution of a small set of real-world examples collected from a small number of users. It then samples from the distribution to produce an unbounded number of synthetic training samples to cover the problem space, allowing one to train an accurate recognition model without incurring great overhead for training data collection.

We demonstrate the benefit of CrossGR by applying it to perform gesture recognition across 10 users and 15 gestures. Our extensive evaluation shows that delivers comparable or even better gesture recognition performance compared to state-of-the-art recognition systems, but using an order of magnitude less training samples collected from the end-users. The result is a new low-cost approach for cross-target gesture recognition with high accuracy.

ACKNOWLEDGMENTS

This work is partially supported by the National Natural Science Foundation of China under grant agreements 61802307, 61772422, and 61872294, and the Shaanxi International Science and Technology Cooperation Program under grant agreements 2020KW-008, 2019KW-004 and 2019KWZ-05, and the ShaanXi Science and Technology Innovation Team Support Project under grant agreement 2018TD-026.

REFERENCES

[1] Mahdi Abavisani, Hamid Reza Vaezi Joze, and Vishal M Patel. 2019. Improving the performance of unimodal dynamic hand-gesture recognition with multimodal training. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 1165–1174.

[2] Fadel Adib and Dina Katabi. 2013. See through walls with WiFi!. In *Proceedings of the ACM SIGCOMM 2013 conference on SIGCOMM*. 75–86.

[3] Fadel Adib, Hongzi Mao, Zachary Kabelac, Dina Katabi, and Robert C Miller. 2015. Smart homes that monitor breathing and heart rate. In *Proceedings of the 33rd annual ACM conference on human factors in computing systems*. 837–846.

[4] Kamran Ali, Alex X Liu, Wei Wang, and Muhammad Shahzad. 2017. Recognizing keystrokes using WiFi devices. *IEEE Journal on Selected Areas in Communications* 35, 5 (2017), 1175–1190.

[5] Sanjeev Arora, Simon S Du, Wei Hu, Zhiyuan Li, and Ruosong Wang. 2019. Fine-grained analysis of optimization and generalization for overparameterized two-layer neural networks. *arXiv preprint arXiv:1901.08584* (2019).

[6] Osbert Bastani, Carolyn Kim, and Hamsa Bastani. 2017. Interpreting blackbox models via model extraction. *arXiv preprint arXiv:1705.08504* (2017).

[7] Liqiong Chang, Jiaqi Lu, Ju Wang, Xiaojiang Chen, Dingyi Fang, Zhanyong Tang, Petteri Nurmi, and Zheng Wang. 2018. SleepGuard: capturing rich sleep information using smartwatch sensing data. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 2, 3 (2018), 1–34.

[8] Bo Chen, Vivek Yenamandra, and Kannan Srinivasan. 2015. Tracking keystrokes using wireless signals. In *Proceedings of the 13th Annual International Conference on Mobile Systems, Applications, and Services*. 31–44.

[9] Kaixuan Chen, Lina Yao, Dalin Zhang, Xiaojun Chang, Guodong Long, and Sen Wang. 2019. Distributionally robust semi-supervised learning for people-centric sensing. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 33. 3321–3328.

[10] 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*. 23–33.

[11] Anmol Sheth David Wetherall Daniel Halperin, Wenjun Hu. [n.d.]. Linux 802.11n CSI Tool. http://dhalperi.github.io/linux-80211n-csitool/faq.html.

[12] Christopher Dondzila and Dena Garner. 2016. Comparative accuracy of fitness tracking modalities in quantifying energy expenditure. *Journal of medical engineering & technology* 40, 6 (2016), 325–329.

[13] Biyi Fang, Nicholas D Lane, Mi Zhang, Aidan Boran, and Fahim Kawsar. 2016. BodyScan: Enabling radio-based sensing on wearable devices for contactless activity and vital sign monitoring. In *Proceedings of the 14th annual international conference on mobile systems, applications, and services*. 97–110.

[14] Biyi Fang, Nicholas D Lane, Mi Zhang, and Fahim Kawsar. 2016. Headscan: A wearable system for radio-based sensing of head and mouth-related activities. In *2016 15th ACM/IEEE International Conference on Information Processing in Sensor Networks (IPSN)*. IEEE, 1–12.

[15] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. 2014. Generative adversarial nets. In *Advances in neural information processing systems*. 2672–2680.

[16] Dan Hendrycks and Kevin Gimpel. 2016. A baseline for detecting misclassified and out-of-distribution examples in neural networks. *arXiv preprint arXiv:1610.02136* (2016).

[17] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A Efros. 2017. Image-to-image translation with conditional adversarial networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 1125–1134.

[18] Wenjun Jiang, Chenglin Miao, Fenglong Ma, Shuochao Yao, Yaqing Wang, Ye Yuan, Hongfei Xue, Chen Song, Xin Ma, Dimitrios Koutsonikolas, et al. 2018. Towards Environment Independent Device Free Human Activity Recognition. In *Proceedings of the 24th Annual International Conference on Mobile Computing and Networking*. ACM, 289–304.

[19] Kanitthika Kaewkannate and Soochan Kim. 2016. A comparison of wearable fitness devices. *BMC public health* 16, 1 (2016), 433.

[20] Hong Li, Wei Yang, Jianxin Wang, Yang Xu, and Liusheng Huang. 2016. WiFinger: talk to your smart devices with finger-grained gesture. In *Proceedings of the 2016 ACM International Joint Conference on Pervasive and Ubiquitous Computing*. 250–261.

[21] Yongsen Ma, Gang Zhou, Shuangquan Wang, Hongyang Zhao, and Woosub Jung. 2018. Signfi: Sign language recognition using wifi. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 2, 1 (2018), 1–21.

[22] Laurens van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-SNE. *Journal of machine learning research* 9, Nov (2008), 2579–2605.

[23] Luke Metz, Ben Poole, David Pfau, and Jascha Sohl-Dickstein. 2016. Unrolled generative adversarial networks. *arXiv preprint arXiv:1611.02163* (2016).

[24] Mehdi Mirza and Simon Osindero. 2014. Conditional generative adversarial nets. *arXiv preprint arXiv:1411.1784* (2014).

[25] Tu Nguyen, Trung Le, Hung Vu, and Dinh Phung. 2017. Dual discriminator generative adversarial nets. In *Advances in Neural Information Processing Systems*. 2670–2680.

[26] Kai Niu, Fusang Zhang, Jie Xiong, Xiang Li, Enze Yi, and Daqing Zhang. 2018. Boosting fine-grained activity sensing by embracing wireless multipath effects. In *Proceedings of the 14th International Conference on emerging Networking EXperiments and Technologies*. 139–151.

[27] Michael Pradel and Koushik Sen. 2018. DeepBugs: A learning approach to name-based bug detection. *OOPSLAs* (2018).

[28] Qifan Pu, Sidhant Gupta, Shyamnath Gollakota, and Shwetak Patel. 2013. Whole-home gesture recognition using wireless signals. In *Proceedings of the 19th annual international conference on Mobile computing & networking*. ACM, 27–38.

[29] 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 *Proceedings of the 16th Annual International Conference on Mobile Systems, Applications, and Services*. 350–361.

[30] Kun Qian, Chenshu Wu, Zimu Zhou, Yue Zheng, Zheng Yang, and Yunhao Liu. 2017. Inferring motion direction using commodity wi-fi for interactive exergames. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems*. 1961–1972.

[31] Alec Radford, Luke Metz, and Soumith Chintala. 2015. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434* (2015).

[32] Stan Salvador and Philip Chan. 2007. Toward accurate dynamic time warping in linear time and space. *Intelligent Data Analysis* 11, 5 (2007), 561–580.

[33] Eric Tzeng, Judy Hoffman, Kate Saenko, and Trevor Darrell. 2017. Adversarial discriminative domain adaptation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 7167–7176.

[34] Raghav H Venkatnarayan, Griffin Page, and Muhammad Shahzad. 2018. Multi-User Gesture Recognition Using WiFi. In *Proceedings of the 16th Annual International Conference on Mobile Systems, Applications, and Services*. ACM, 401–413.

[35] Aditya Virmani and Muhammad Shahzad. 2017. Position and orientation agnostic gesture recognition using wifi. In *Proceedings of the 15th Annual International Conference on Mobile Systems, Applications, and Services*. ACM, 252–264.

[36] Tran Huy Vu, Archan Misra, Quentin Roy, Kenny Choo Tsu Wei, and Youngki Lee. 2018. Smartwatch-based early gesture detection 8 trajectory tracking for interactive gesture-driven applications. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 2, 1 (2018), 1–27.

[37] Huogen Wang, Pichao Wang, Zhanjie Song, and Wanqing Li. 2017. Large-scale multimodal gesture recognition using heterogeneous networks. In *Proceedings of the IEEE International Conference on Computer Vision Workshops*. 3129–3137.

[38] Wei Wang, Alex X Liu, and Muhammad Shahzad. 2016. Gait recognition using wifi signals. In *Proceedings of the 2016 ACM International Joint Conference on Pervasive and Ubiquitous Computing*. 363–373.

[39] Wei Wang, Alex X Liu, Muhammad Shahzad, Kang Ling, and Sanglu Lu. 2015. Understanding and modeling of wifi signal based human activity recognition. In *Proceedings of the 21st annual international conference on mobile computing and networking*. ACM, 65–76.

[40] Wei Wang, Alex X Liu, Muhammad Shahzad, Kang Ling, and Sanglu Lu. 2017. Device-free human activity recognition using commercial WiFi devices. *IEEE Journal on Selected Areas in Communications* 35, 5 (2017), 1118–1131.

[41] Yan Wang, Jian Liu, Yingying Chen, Marco Gruteser, Jie Yang, and Hongbo Liu. 2014. E-eyes: device-free location-oriented activity identification using fine-grained wifi signatures. In *Proceedings of the 20th annual international conference on Mobile computing and networking*. ACM, 617–628.

[42] Yaxiong Xie, Zhenjiang Li, and Mo Li. 2018. Precise power delay profiling with commodity Wi-Fi. *IEEE Transactions on Mobile Computing* 18, 6 (2018), 1342–1355.

[43] Jason Yosinski, Jeff Clune, Anh Nguyen, Thomas Fuchs, and Hod Lipson. 2015. Understanding neural networks through deep visualization. *arXiv preprint arXiv:1506.06579* (2015).

[44] Lantao Yu, Weinan Zhang, Jun Wang, and Yong Yu. 2017. Seqgan: Sequence generative adversarial nets with policy gradient. In *Thirty-First AAAI Conference on Artificial Intelligence*.

[45] Yinggang Yu, Dong Wang, Run Zhao, and Qian Zhang. 2019. RFID based real-time recognition of ongoing gesture with adversarial learning. In *Proceedings of the 17th Conference on Embedded Networked Sensor Systems*. 298–310.

[46] Ming Zeng, Le T Nguyen, Bo Yu, Ole J Mengshoel, Jiang Zhu, Pang Wu, and Joy Zhang. 2014. Convolutional neural networks for human activity recognition using mobile sensors. In *6th International Conference on Mobile Computing, Applications and Services*. IEEE, 197–205.

[47] Yunze Zeng, Parth H Pathak, and Prasant Mohapatra. 2016. WiWho: wifi-based person identification in smart spaces. In *2016 15th ACM/IEEE International Conference on Information Processing in Sensor Networks (IPSN)*. IEEE, 1–12.

[48] Fusang Zhang, Kai Niu, Jie Xiong, Beihong Jin, Tao Gu, Yuhang Jiang, and Daqing Zhang. 2019. Towards a diffraction-based sensing approach on human activity recognition. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 3, 1 (2019), 1–25.

[49] Jie Zhang, Zhanyong Tang, Meng Li, Dingyi Fang, Petteri Nurmi, and Zheng Wang. 2018. CrossSense: Towards Cross-Site and Large-Scale WiFi Sensing. In *Proceedings of the 24th Annual International Conference on Mobile Computing and Networking*. ACM, 305–320.

[50] Liang Zhang, Guangming Zhu, Peiyi Shen, Juan Song, Syed Afaq Shah, and Mohammed Bennamoun. 2017. Learning spatiotemporal features using 3dcnn and convolutional lstm for gesture recognition. In *Proceedings of the IEEE International Conference on Computer Vision Workshops*. 3120–3128.

[51] 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 *Proceedings of the 17th Annual International Conference on Mobile Systems, Applications, and Services*. ACM, 313–325.