One is Enough: Enabling One-shot Device-free Gesture Recognition with COTS WiFi

Leqi Zhao[1,2*], Rui Xiao[1*], Jianwei Liu[1], and Jinsong Han[1,2]

[1]Zhejiang University, China
[2]Key Laboratory of Blockchain and Cyberspace Governance of Zhejiang Province, China

Abstract—**In recent years, WiFi-based gesture recognition (WGR) has gained popularity due to its privacy-preserving nature and the wide availability of WiFi infrastructure. However, existing WGR systems suffer from scalability issues, i.e., requiring extensive data collection and re-training for each new gesture class. To address these limitations, we propose *OneSense*, a one-shot WiFi-based gesture recognition system that can efficiently and easily adapt to new gesture classes. Specifically, we first propose a data enrichment approach based on the law of signal propagation in physical world to generate virtual gestures, enhancing the diversity of the training set without extra overhead of real sample collection. Then, we devise an aug-meta learning (AML) framework to enable efficient and scalable few-short learning. This framework leverages two pre-training stages (i.e., aug-training and meta-training) to improve the model's feature extraction and generalization abilities, and ultimately achieves accurate one-shot gesture recognition through fine-tuning. Experimental results demonstrate that *OneSense* achieves 93% one-shot gesture recognition accuracy, which outperforms the state-of-the-art approaches. Moreover, it maintains high recognition accuracy when facing new environments, user locations, and user orientations. Furthermore, the proposed AML framework reduces 86%+ pre-training latency compared to conventional meta-learning method.**

Index Terms—**WiFi Sensing, Gesture Recognition, Few-shot Learning**

I. Introduction

In the past decade, WiFi-based gesture recognition (WGR) has attracted increasing attention owing to its multiple irresistible properties, such as visual privacy preserving, robustness to occlusion, and wide deployment of WiFi facilities [1]–[4]. Existing WGR systems typically rely on learning models to associate channel state information (CSI) with gesture classes. They demonstrate high gesture recognition accuracy, yet, also have two major flaws on the system scalability, which hinder their landings in real world.

On the one hand, to achieve accurate gesture prediction, the learning model generally needs to derive knowledge from a dataset containing the CSI samples of every gesture class. As each class should provide dozens or even hundreds of samples, the data collection is significantly time-consuming and resource-intensive. On the other hand, traditional models circumscribe the gesture classes at the beginning of the system implementation. If one wants the system to recognize new gestures, the system necessitates a re-training for the whole model. This introduces lots of extra computational overhead. As the demand for gesture categories continues to increase [5], the above flaws make existing systems struggle to fulfill flexible gesture recognition tasks.

Recently, few-shot learning techniques show potential in addressing the drawbacks mentioned above. Previous works [6]–[8] in computer vision (CV) field have demonstrated the feasibility of few-shot image recognition via *meta-learning*. The key idea behind meta-learning is to build a general model in advance from a large number of similar few-shot tasks. This model can then quickly adapt to new tasks with only a limited number of labeled samples. However, applying meta-learning to WiFi-based gesture recognition is non-trivial due to the following challenges. (1) The training of the general model requires a diverse dataset that contains a large quantity of gesture classes (termed as seen classes). The developer has to collect massive samples to form such a diverse dataset. This consumes a lot of time and manpower. (2) The meta-learning mechanism would construct copious few-shot tasks for pre-training, which usually demands far more computational and time overhead than normal supervised learning. (3) Conventional meta-learning does not support varying numbers of gesture classes. For example, once the general model was well-trained, it cannot be used to recognize one more new gesture class (termed as unseen class), unless re-training the entire general model. This hampers the system's capability in handling evolving sensing requirements.

By addressing these challenges, we propose a novel one-shot WiFi-based gesture recognition system named *OneSense*. It frees up lots of manpower, time, as well as computational overhead for both developers and users. The developer first needs to collect a small number of real samples, based on which a large number of virtual samples are generated. Combing the real and virtual samples, the developer can quickly train a general model. By fine-tuning the general model with only one real sample for each class, the user can achieve accurate one-shot gesture recognition.

Specifically, we first design a data enrichment algorithm to increase the diversity of the training dataset, eliminating the reliance on extensive real-world data collection. In particular, developers only need to collect a few real-world samples for several gesture classes, and leverage the data enrichment algorithm to generate abundant new gestures/samples (namely

*Both authors contributed equally to this research. Jianwei Liu and Jinsong Han are corresponding authors.

virtual gestures/samples) based on the law of signal propagation in physical world. This algorithm helps construct a diverse dataset for training the general model with much less time and manpower than real collection.

Then, we devise a novel one-shot learning framework called *aug-meta learning (AML)*, which incorporates the advantages of both normal supervised learning and meta-learning. The framework consists of two pre-training stages, i.e. *aug-training* and *meta-training*. In the aug-training stage, the model undergoes normal supervised learning using the virtual samples, gaining the ability of deep feature extraction. In the meta-training stage, the AML framework will perform classical meta-learning to adapt to few-shot scenarios, and generate a general model. The combination of these two pre-training stages accelerates the convergence of the model, greatly reducing the computational and time overhead.

Finally, the system is customized to suit different gesture recognition tasks by simply fine-tuning the pre-trained general model with only one sample for each class. The number of gesture classes is tunable in this stage, which means that, once pre-trained, *OneSense* is capable of meeting evolving sensing requirements with little overhead.

We build a prototype of *OneSense* and conduct extensive experiments to evaluate its performance in four real environments. The results indicate that *OneSense* can achieve a high one-shot recognition accuracy of 93%, outperforming existing WiFi-based few-shot gesture recognition approaches. Robustness study demonstrates that the recognition performance of *OneSense* remains satisfactory when facing varied environments, user locations, and user orientations. Moreover, the proposed AML framework can reduce over 86% pre-training time cost compared to conventional meta-learning.

The contributions of this paper are summarized as follows:

- We propose a novel scalable one-shot WiFi-based gesture recognition system, namely *OneSense*. It only requires the user to collect one real sample for each new class.
- We design a data enrichment algorithm based on signal propagation law to expand the training dataset, which significantly reduces the manpower and time overhead of real data collection.
- We propose AML framework to enable efficient and scalable few-shot learning. This framework is promising to be applied to many sensing tasks or even fields like CV.
- We conduct extensive experiments in real environments. The results demonstrate that *OneSense* can achieve 93% one-shot gesture recognition accuracy. Meanwhile, *OneSense* is robust against environment, user location, and user orientation variations.

II. Primer

We achieve few-shot gesture recognition using WiFi CSI. This section first presents some preliminary knowledge on WiFi CSI and then introduces the basics of few-shot learning.

A. WiFi CSI

Current WiFi-based sensing techniques dominantly perform sensing tasks by extracting CSI from WiFi packets [9]. WiFi CSI is the channel frequency response of each OFDM subcarrier, describing how WiFi signals propagate from the transmitter to the receiver after experiencing amplitude attenuation, phase shift, and adding noise at physical layer [10]. Each CSI entry with carrier frequency f_c at time t can be formulated as:

$$H(f_c,t)=\sum_{k=1}^{K}\alpha_k(t)e^{-j2\pi f_c\tau_k(t)}+N \tag{1}$$

where K is the number of multipaths. α_k and τ_k are the amplitude attenuation factor and propagation delay for the k-th path, respectively. N is the additive white Gaussian noise.

These K paths can be divided into *static ones* and *dynamic ones*. The static paths include the direct propagation from the transmitter to the receiver and the reflection on static objects in the environment. Dynamic paths refer to the paths reflected by the moving object. Correspondingly, each CSI entry can also be divided into static component $H_s(f_c)$ and dynamic component $H_d(f_c,t)$. Taking into consideration the extra phase offsets caused by the hardware and software errors, the estimated raw CSI entry can be formulated as:

$$H(f_c,t)=(H_s(f_c)+H_d(f_c,t))e^{j\theta(f_c,t)}+N \tag{2}$$

where $e^{j\theta(f_c,t)}$ is the random extra phase offset including timing alignment offset, sampling frequency offset, as well as carrier frequency offset.

In the scenario of human gesture recognition, the moving human body would induce variations in the amplitude and phase of the CSI multipath channels, especially the dynamic ones. Therefore, we can extract gesture-relevant dynamic features from CSI to achieve WiFi-based gesture recognition.

B. Few-shot Learning

Generally, WiFi-based gesture recognition systems leverage supervised learning to map the features extracted from CSI into gesture classes. This requires the user to collect a large number of samples for each class, consuming lots of human efforts. To solve this problem, we adopt few-shot learning (FSL) techniques to reduce the data collection overhead. In FSL like meta-learning [11], the learning problem is typically divided into a series of N-way K-shot tasks $\{T_i\}_{i=1}^{I}$, where N is the number of classes, K represents the number of samples available to learn from for each class, and I denotes the total number of tasks in this series. Each task T_i consists of a support set S_i and a query set Q_i. The support set S_i provides a limited number of labeled samples to train the model, while the query set Q_i is used to evaluate the learning model's performance after training on S_i. The support set S_i is composed of $N \times K$ samples, where N classes are randomly selected from the dataset, and K samples are extracted for each selected class. The query set Q_i also contains samples from the same N classes as in S_i, but with different instances. By organizing the learning process into these tasks and training

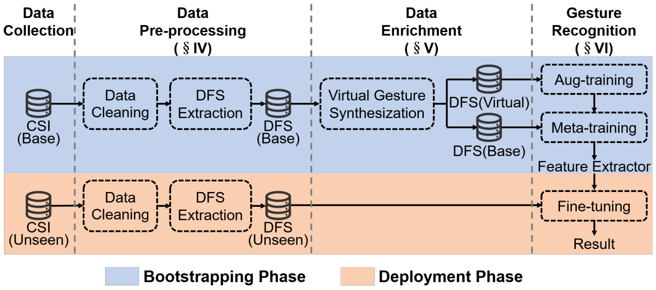


Fig. 1: Architecture of *OneSense*.

the model on different combinations of classes and samples, few-shot learning enables the model to generalize and make accurate predictions on new tasks with limited labeled data.

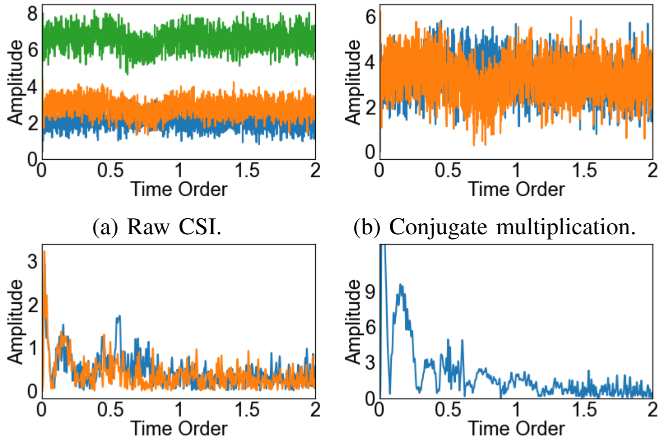
In this work, we aim at minimizing the data collection overhead, i.e., achieving one-shot gesture recognition, with meta-learning technique. To further reduce the computational overhead induced by conventional meta-learning method, we propose a novel few-shot learning framework named AML.

III. System Overview

We propose a one-shot WiFi-based gesture recognition system, namely *OneSense*. As shown in Fig. 1, *OneSense* mainly contains four modules: data collection, data pre-processing, data enrichment, and gesture recognition. The use of *OneSense* can be divided into two phases: *bootstrapping phase* (developer edge) and *deployment phase* (user edge). Our key idea is that, the developers pre-train a recognition model based on the pre-collected samples (base gestures) as well as the virtual ones to bootstrap the system. Then, on the user edge, only a few labeled samples are required to fine-tune the pre-trained model for deployment on the customized task (unseen gestures).

Bootstrapping phase. In this phase, *OneSense* aims to pre-train a feature extractor that forms the foundation for one-shot recognition. Specifically, *OneSense* first collects a batch of CSI samples of seen gestures as base dataset in the data collection module. Then, the data pre-processing module removes gesture-irrelevant components like noise from raw CSI measurements. With the clean CSI, *OneSense* extracts Doppler frequency shift (DFS) as environment-independent gesture features. Thereafter, in the data enrichment module, *OneSense* leverages a virtual gesture construction algorithm to generate a large quantity of samples of virtual gestures based on the base dataset. Finally, in the gesture recognition module, *OneSense* trains a feature extractor using both the virtual dataset and base dataset via AML.

Deployment phase. *OneSense* obtains an accurate classifier for unseen gestures in this phase. To be specific, *OneSense* first gets only one sample for each unseen gesture in the data collection module. Then, the collected samples undergo the same pre-processing as the bootstrapping phase. After that, *OneSense* utilizes the feature extractor pre-trained in the bootstrapping phase, and attaches a classifier after it. *OneSense* fine-tunes the classifier using the pre-processed samples of unseen gestures, obtaining a model with accurate one-shot recognition ability for unseen gestures.



(a) Raw CSI.

(b) Conjugate multiplication.

(c) Filtering.

(d) PCA.

Fig. 2: Effectiveness of signal pre-processing.

IV. Data Pre-processing

This section describes how *OneSense* removes gesture-irrelevant components from raw CSI measurements and extracts environment-independent DFS as gesture features. For clarity, we take only one subcarrier from each antenna as an example in Fig. 2.

A. Data Cleaning

As mentioned in Sec. II-A, raw CSI measurements (Fig. 2(a)) contain many components irrelevant to the gesture information, such as phase offset, static components, and noise. These components could degrade the gesture recognition performance. To suppress their impacts, we apply a series of signal processing techniques to clean the raw CSI, including conjugate multiplication [12], frequency-based filtering, and principal component analysis (PCA) [13].

Conjugate multiplication. The slight out-of-sync between the transmitter and receiver would introduce time-varied random phase offset $e^{j\theta(f_c,t)}$ (Eq. 2). Fortunately, since the antennas on the same receiver share the same RF oscillator, their phase offsets can be considered consistent. Based on this characteristic, we can eliminate such phase offset by performing conjugate multiplication between the CSI of two antennas on the same receiver, as shown in Fig. 2(b).

Frequency-based filtering. In addition to the phase offset, the received CSI also contains static components and white Gaussian noise. To eliminate their influences, we perform high-pass filtering (2Hz cut-off frequency) to remove low-frequency components caused by static paths on one hand, and then conduct low-pass filtering (60Hz cut-off frequency) to erase high-frequency noise on the other. As shown in Fig. 2(c), the filtered CSI traces become more smooth.

PCA. Ultimately, we perform PCA on the filtered CSI and extract the first principal component. This not only makes the gesture-related features in the CSI more prominent, but also removes some remaining noise. It can be observed from

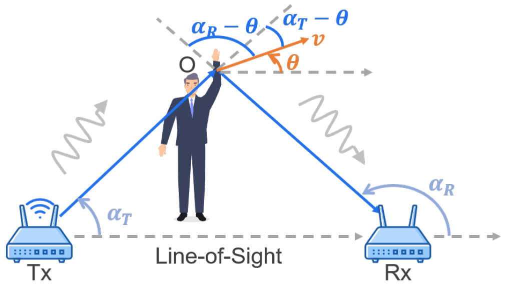


Fig. 3: WiFi signal propagation model in physical world.

Fig. 2(d) that the CSI after PCA is clean. The envelope, i.e., the variation trend, of the CSI profile is clearer, allowing *OneSense* to extract representative and high-quality gesture features.

B. DFS Extraction

Due to the multipath effect, the CSI measurements are related to not only the dynamics of the human body, but also the surrounding environments. In this case, The CSI profiles of the same gesture collected in different environments may differ. Further, the gesture recognition model built in one environment could perform inadequately in another. To address this issue, we opt to extract DFS from CSI as gesture features. Since DFS is environment-independent, the model built upon DFS can perform well across environments. In the following, we will show how to extract DFS from WiFi CSI according to the Doppler effect [14].

A scenario of WiFi-based human gesture recognition is shown in Fig. 3. If we regard the target user O as a point, due to Doppler Effect, the motion of O will lead to a frequency change between the transmitter Tx and the receiver Rx. Such a variation in frequency is called DFS, which can be calculated by:

$$f_D = -f_0 \frac{v}{c}(cos(\theta - \alpha_R) + cos(\theta - \alpha_T)) \tag{3}$$

where f_0 is the frequency of the WiFi signal transmitted by Tx, v is the speed of the target user, c is the speed of light, α_T is the angle of departure (AoD), and α_R is the angle of arrival (AoA).

However, in reality, a human body cannot be simply seen as a point, especially in near-field scenarios where the distance between the human and the transmitter/receiver is small. In fact, different body parts would indice different velocity components, and consequently, different DFS. To deal with this problem, we introduce Doppler spectrogram to represent the intensity of DFS components over time, which covers the whole body. Doppler spectrogram can be estimated from CSI measurements by time-frequency analysis techniques such as short-time Fourier transform (STFT). Hence, after data cleaning, we first employ STFT to obtain the Doppler spectrogram from CSI, and then retain the part of the spectrogram that mostly reflects the dynamics of the human body as the input of the subsequent learning model.

Specifically, let $x(t)$ be the time-series CSI data and $w(t)$ be a window function in STFT. The Doppler spectrogram at a particular frequency f and time t can be obtained by calculating the square of the STFT magnitude, which is given by:

$$S(f,t) = \left| \int_{-\infty}^{\infty} x(\tau)w(t-\tau)\mathrm{e}^{-2\pi i f \tau} d\tau \right|^2 \tag{4}$$

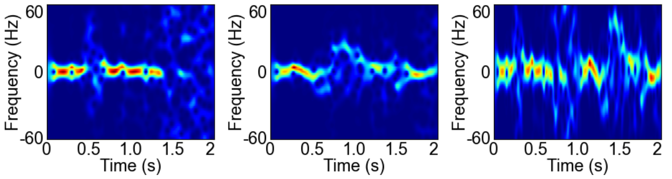
After that, we retain the part that best represents the Doppler shift in the spectrogram, i.e., the part within the frequency range of [-60Hz, 60Hz] [5]. The ultimate Doppler spectrogram, as shown in Fig. 4(a) and (b), provides valuable insights into the movements of the human body.

V. Data Enrichment

As mentioned in Sec. II-B, users need to prepare a dataset with plentiful gesture classes to pre-train the learning model, when adopting meta-learning for one-shot recognition. Apparently, collecting real gesture samples in physical world is laborious. To tackle this problem, Xiao et al. [5] propose to generate virtual gestures by simulating the rotation of real gestures in a two-dimensional plane, i.e., changing the orientation of the real gestures. This indeed increases the size of the support set, yet, does not essentially introduce new gesture classes, as a virtual gesture remains the same as the real gesture used to generate it. With this virtual gesture generation scheme, the model may mistake the same class of gestures with slightly different orientations for different classes.

In this section, we propose a novel virtual gesture synthesization method to generate new gesture classes that have not been explicitly observed during data collection. A virtual gesture sample refers to a synthesized gesture sample that retains the temporal characteristics and movement patterns of multiple source gestures while introducing new combinations of gestures. For example, if we have a source gesture ‘L’ and another source gesture ‘I’, then we can connect them to get a virtual sample ‘⊔’. However, achieving such a synthesis is challenging as: (1) Directly splicing two source samples in the temporal domain would make the duration of the generated sample inconsistent with those of source samples. (2) If we discard half of the elements of each source sample and then concatenate, the duration of the newly generated sample will not be abnormal, but the gesture information will be lost and the virtual sample will not conform to the signal propagation law in the physical world. To overcome this challenge, we come up with a solution that does not prolong the duration of new samples while conforming to the laws of physics. The key idea is that we can accelerate the samples of two real gesture classes and combine them to get a new class based on the signal propagation model.

Gesture acceleration. To shorten the duration of the sample while maintaining its inner gesture characteristics, we accelerate gesture samples based on the physical signal propagation law. Consider a source sample of a predefined gesture, if we accelerate it to take only $\frac{1}{n}$ of the original time, then the position of the moving target at time t after acceleration is the same as that at time nt before acceleration. Therefore, we have the following relations: $s_{acc}(t) = s(nt)$, $\theta_{acc}(t) = \theta(nt)$,



(a) Drawing '1'

(b) Drawing '2'.

(c) Virtual gesture.

Fig. 4: Generating Doppler spectrograms of virtual gesture (c) by combining real ones (a) and (b).

$\alpha_{R_{acc}}(t) = \alpha_R(nt)$, $\alpha_{T_{acc}}(t) = \alpha_T(nt)$, where s, θ, α_R, α_T denote the passed distance, moving direction, AoA and AoD of the source sample, respectively; s_{acc}, θ_{acc}, $\alpha_{R_{acc}}$, $\alpha_{T_{acc}}$ denote those corresponding measurements after acceleration, respectively. Then, the velocity at time t after acceleration can be calculated by:

$$v_{acc}(t) = \frac{\mathrm{d}s_{acc}(t)}{\mathrm{d}t} = \frac{\mathrm{d}s(nt)}{\mathrm{d}t} = n\frac{\mathrm{d}s(t)}{\mathrm{d}t} = n \cdot v(nt) \quad (5)$$

Combining with Eq. 3, we obtain the DFS of the accelerated gesture at time t:

$$\begin{aligned} f_{D_{acc}}(t) &= -f_0\frac{v_{acc}(t)}{c}(cos(\theta_{acc}(t) - \alpha_{R_{acc}}(t)) \\ &+ cos(\theta_{acc}(t) - \alpha_{T_{acc}}(t))) \\ &= -f_0\frac{n \cdot v(nt)}{c}(cos(\theta(nt) - \alpha_R(nt)) \\ &+ cos(\theta(nt) - \alpha_T(nt))) \\ &= n \cdot f_D(nt) \end{aligned} \quad (6)$$

Denote the Doppler spectrogram of the source sample as $S(f,t)$, where $f \in [-F,F]$, $t \in [0,T]$. Then, for the accelerated gesture, the Doppler spectrogram can be obtained as: $S_{acc}(f,t) = S(\frac{f}{n}, nt)$, where $f \in [-nF, nF]$, $t \in [0, \frac{T}{n}]$.

Virtual gesture generation. We then combine the accelerated samples to construct virtual samples in time length of the source ones. Consider two samples of any two different source gestures A and B, we can generate the Doppler spectrogram sample of the virtual gesture A+B as:

$$S_{A+B}(f,t) = \begin{cases} S_A\left(\frac{f}{2}, 2t\right) & 0 \le t < \frac{T}{2} \\ S_B\left(\frac{f}{2}, 2t - T\right) & \frac{T}{2} \le t \le T \end{cases} \quad (7)$$

The above process describes how to generate one virtual gesture with two real gestures. In fact, this method can be trivially extended to more than two real gestures. Theoretically, users can generate infinite virtual gesture classes as well as corresponding virtual samples to enrich the training set. Fig. 4(c) shows the Doppler spectrograms of a virtual gesture synthesized by two source gestures Fig. 4(a) and (b), demonstrating the effectiveness of our virtual gesture generation approach.

VI. Aug-Meta Learning

We aim at leveraging meta-learning to achieve one-shot unseen gesture recognition. Nevertheless, we find that the pre-training stage of the meta-learning would introduce a lot of time cost and computational overhead. In this section, on the basis of meta-learning, we design a new few-shot learning framework named AML. It enables efficient and fast pre-training while maintaining high one-shot gesture recognition performance.

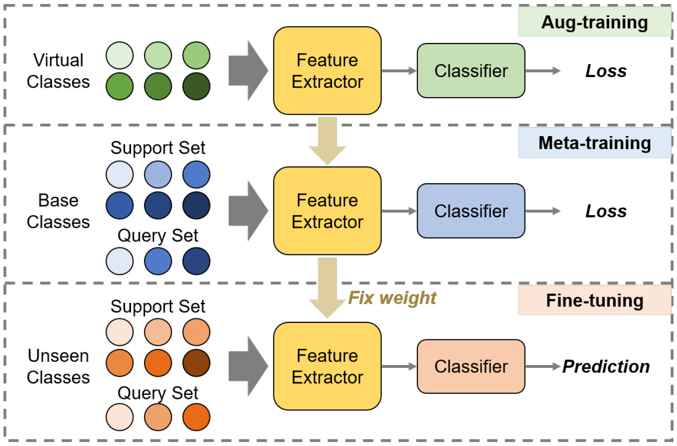


Fig. 5: Aug-meta learning framework.

TABLE I: Details of feature extractor architecture.

Layer	Type	Channel	Kernel Size	Stride
conv1	conv2d+BN+ReLU	16	3×5	(2,2)
conv2	conv2d+BN+ReLU	32	4×4	(2,2)
conv3	conv2d+BN+ReLU	64	4×4	(2,2)

As shown in Fig. 5, our AML framework consists of three stages, namely *aug-training*, *meta-training*, and *fine-tuning*. The key idea is that: AML first employs the *aug-training* stage to empower the model with deep feature extraction ability through normal supervised learning over virtual gestures. Then, in the *meta-training* stage, AML pre-trains the model with original meta-learning technique over real collected gestures to enhance the model's generalization capability and adaptability to few-shot scenarios. In the *fine-tuning* stage, AML fine-tunes the pre-trained model with samples of unseen gestures to enhance its specialization towards recognizing these previously unseen gestures. The learning model mentioned above is composed of a feature extractor followed by a classifier. The feature extractor contains three convolutional layers, as shown in Tab. I. The classifier consists of a single fully-connected layer. It is attached to the last convolutional layer of the feature extractor, mapping the extracted deep features to the confidence scores for different gesture classes. The details of the three stages are as follows.

A. *Aug-training*

In the aug-training stage, we pre-train a feature extractor f_θ and a classifier c_ω using the generated virtual dataset D_{virtual} via classical supervised learning. We start pre-training by randomly initializing the parameters of f_θ and c_ω. Then, we optimize the parameter sets θ and ω by minimizing the following loss function:

$$\mathcal{L}_{D_{\text{virtual}}}(\theta,\omega) = \frac{1}{|D_{\text{virtual}}|}\sum_{(x,y)\in D_{\text{virtual}}} l(c_\omega(f_\theta(x)), y) \quad (8)$$

TABLE II: Details of the gesture classes considered in the evaluation. (CCW means counterclockwise.)

Base Gestures	Draw ’1’	Draw ’2’	Draw ’3’	Draw ’4’	Draw ’5’
	Draw ’6’	Draw ’7’	Draw ’8’	Draw ’9’	Draw ’0’
	Draw ’0’ (CCW)	Draw ’U’	Draw ’U’ reverse	Draw ’N’	Draw ’N’ reverse
	Swing left and right	Swing right and left	Draw Rectangle	Draw Rectangle (CCW)	Dig
Unseen Gestures	Push and Pull	Sweep	Slide	Clap	Draw zig-zag
	Draw Triangle	Draw ’a’	Draw ’b’	Draw ’c’	Draw ’d’
	Draw ’e’	Draw ’f’	Draw ’g’	Draw ’h’	Draw ’i’
	Draw ’j’	Draw ’k’	Draw ’l’	Draw ’m’	Draw ’n’

where l represents the cross-entropy loss [15]; x and y denote a gesture sample and its label, respectively.

After the above pre-training, the feature extractor will possess the ability of deep feature extraction. As the number of virtual gesture classes may be different from that of the dataset used for meta-learning in the next stage, the classifier c_ω will be discarded after aug-training.

B. Meta-training

In the meta-training stage, we employ FSL techniques to sample a set of n-way 1-shot tasks $\{T_i\}_{i=1}^{I}$ from the real collected dataset, i.e., base dataset D_{base}, where the integer n can be selected from the interval $[2, N_{\text{base}}]$ (N_{base} is the number of classes in D_{base}). We start training by loading the parameters of the feature extractor f_θ pre-trained in the previous stage, and randomly initializing a new classifier c_ϕ. For each task $T_i = (S_i, Q_i)$, we use the support set S_i to optimize the parameters ϕ by minimizing the loss $\mathcal{L}_{S_i}(\theta, \phi)$. Similar to Eq. 8, $\mathcal{L}_{S_i}(\theta, \phi)$ is calculated as follows:

$$\mathcal{L}_{S_i}(\theta, \phi) = \frac{1}{|S_i|} \sum_{(x,y) \in S_i} l(c_\phi(f_\theta(x)), y) \qquad (9)$$

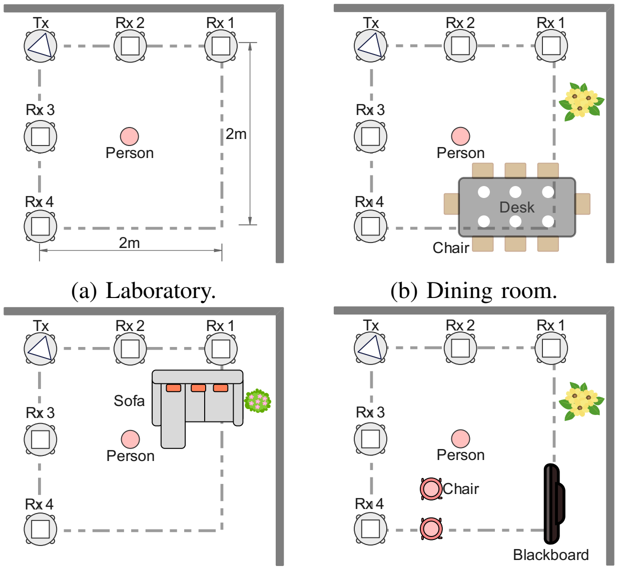
During the above optimization process, we do not update the model parameters directly. Instead, we record the optimized parameters as ϕ_i for each task T_i. Then, we calculate the loss on the query set Q_i using the optimized parameters ϕ_i, which is denoted as $\mathcal{L}_{Q_i}(\theta, \phi_i)$.

Once we have completed training on all the tasks in $\{T_i\}_{i=1}^{I}$, we proceed to adapt the parameter sets θ and ϕ of the model. This is done by minimizing the accumulated loss $\sum_{i=1}^{I} \mathcal{L}_{Q_i}(\theta, \phi_i)$.

In this stage, the feature extractor is updated to learn better representations from real-world samples and become more capable of handling few-shot scenarios. The optimized feature extractor will be directly used in the subsequent stage, while the classifier will be discarded again.

C. Fine-tuning

In the final stage, we generate a N_{unseen}-way 1-shot support set S from dataset D_{unseen}, where N_{unseen} is the number of classes in D_{unseen}. This support set consists of only one labeled sample for each unseen class. We first load the pre-trained feature extractor f_θ from the previous stage and fix its parameters. Then, we attach a new classifier c_ψ to the tail of f_θ and fine-tune it over the one-shot support set S. After fine-tuning, the feature extractor and classifier work together to recognize unseen gestures with high accuracy.



(a) Laboratory. (b) Dining room.

(c) Living room. (d) Office.

Fig. 6: Experiment setup in four environments.

VII. Evaluation

This section presents the real-world implementation and evaluation results of *OneSense*.

Experiment setup. We conduct experiments in four environments, including a laboratory, a dining room, a living room, and an office, as shown in Fig. 6. The sensing area is a $2m \times 2m$ square. We use one transmitter and four receivers, all of which are commercial off-the-shelf (COTS) laptops equipped with Intel 5300 network interface cards (NICs). The transmitter (with one antenna) sends WiFi packets at a rate of 1000 packets/s. Each receiver is set to monitor mode with three antennas. Linux 802.11n CSI Tool [16] is installed to extract CSI measurements from WiFi packets. The signals lie in 5.54GHz.

Data collection. We recruit 10 participants to perform gestures in our experiments, including six males and four females. We define 40 gestures as shown in Tab. II, 20 classes of which are used as base gestures, and the remaining 20 are regarded as unseen gestures. Each base gesture is performed at least 30 times. We totally collect over 2900 gesture samples. In the default setting, we use 20 base gestures and 6 unseen gestures (‘push and pull’, ‘sweep’, ‘slide’, ‘clap’, ‘draw zig-zag’, and ‘draw triangle’) performed in the laboratory environment for evaluation. In K-shot recognition, K samples are randomly selected for fine-tuning and the remaining samples are used for test.

Metric. We define accuracy to quantify the performance of gesture recognition. Accuracy represents the probability that a sample is correctly recognized. It can be calculated by: $accuracy = \frac{N_{cor}}{N_{all}}$, where N_{cor} and N_{all} denote the number of correctly classified samples and the number of all tested samples, respectively.

A. Overall Performance

We first compare *OneSense* with two state-of-the-art few-shot gesture recognition systems, and then assess the effects of our AML framework, data enrichment approach, number of base classes, and number of unseen classes.

Comparison with state-of-the-art works. We compare *OneSense* with two state-of-the-art WiFi-based few-shot gesture recognition approaches: OneFi [5] and WiGr [17]. OneFi leverages data augmentation and transfer learning techniques to recognize unseen gestures with only a few samples. WiGr employs a modified prototypical network for few-shot recognition. Fig. 7 shows the accuracies of OneFi, WiGr, and *OneSense* under one-shot, three-shot, and five-shot settings. In one-shot recognition, *OneSense* achieves an accuracy of 93.0%, surpassing OneFi and WiGr, which achieve 84.2% and 81.7%, respectively. *OneSense* also demonstrates superiority under three-shot and five-shot settings. Moreover, *OneSense* can easily adapt to variable unseen gesture classes, while WiGr lacks the flexibility to change the number of unseen gesture classes once its model is well-trained. Thus, *OneSense* outperforms OneFi and WiGr, showing great user-friendliness with its impressive ability in recognizing gestures using only one training sample for each unseen gesture class.

Effect of AML framework. To demonstrate the advantages of our AML framework, we compare *OneSense* with two baselines: (1) retaining only the first pre-training stage, i.e., aug-training; (2) retaining only the second pre-training stage, i.e., meta-learning. Fig. 8 illustrates the accuracies under one-shot, three-shot, and five-shot settings. It can be found that, in the one-shot setting, *OneSense* achieves higher accuracy than the two baselines by a remarkable margin. For three-shot and five-shot recognition, *OneSense* still outperforms the baselines. This indicates that both the aug-training and meta-training stages effectively improve the few-shot recognition performance.

Effect of data enrichment. To reduce the manpower and time consumed to collect real data, we propose a novel data enrichment method, synthesizing virtual gestures to augment the dataset without manual collection. To evaluate the effect of our data enrichment method, we vary the number of generated virtual gesture classes from 0 to 380 in step of 20. Fig. 9 presents the accuracies of one-shot, three-shot, and five-shot recognition. The accuracies exhibit a noticeable increase as the number of virtual gesture classes grows. When the number of classes reaches 40, the accuracy curves become flat, indicating that the accuracies tend to converge. These results suggest that: (1) Data enrichment through virtual gesture generation is beneficial for improving few-shot recognition performance on new gestures. (2) With only a small number of virtual gesture classes, *OneSense* can achieve outstanding recognition accuracies. This means that users only need to collect the samples of a few real gesture classes, based on which *OneSense* can generate sufficient virtual gesture classes.

Effect of number of base classes. As the number of base classes determines the size of training datasets and reflects the data collection overhead for pre-training, understanding how it affects the recognition accuracy is crucial for optimizing the system's performance. To investigate its effect, we vary the number of base classes from 0 to 20 in step of 1 and recalculate the accuracies under one-shot, three-shot, and five-shot settings. As shown in Fig. 10, when the number of base classes is below six, the recognition accuracies exhibit substantial improvement with each additional base class. For instance, in one-shot setting, the accuracy increases from 57.2% with 2 base classes to 93.7% with 6 base classes. Once *OneSense* employs over six base classes, the accuracies become stable. From the above analysis, we conclude that: (1) Involving base classes in pre-training indeed improves the recognition performance of *OneSense*. (2) *OneSense* can achieve satisfactory few-shot recognition performance with a relatively small number of base classes, which further reduces the overhead of data collection and model pre-training.

Effect of number of unseen classes. In practical applications, users may require recognition of varying numbers of gesture classes. We evaluate this scalability of *OneSense* by varying the number of unseen classes from 2 to 20 in step of 1. The results under one-shot, three-shot, and five-shot settings are presented in Fig. 11. As expected, the accuracies show a decreasing trend with an increasing number of unseen classes. Nevertheless, even with a large number of unseen classes, the accuracies remain high. For instance, the one-shot recognition accuracy stays above 85% with 9 unseen classes. When dealing with 20 unseen classes, *OneSense* can still achieve accuracies of 72.8%, 87.5% and 91.2% in one-shot, three-shot and five-shot recognition, respectively. Therefore, *OneSense* can effectively adapt to new gestures without significant re-training efforts and maintain excellent performance even with a large number of classes.

B. Effect of Number of Receivers

The number of receivers is a crucial hardware configuration that may affect the system's performance. While having more receivers generally provides CSI with more comprehensive information, it also introduces larger resource overhead. In this section, we explore the effect of the number of receivers on the performance of *OneSense* by varying it from 1 to 4 in step of 1. The resulting accuracies under one-shot, three-shot, and five-shot settings are depicted in Fig. 12. We observe that the accuracies do not vary obviously with the changes in the number of receivers. Even when employing only one receiver, the accuracy remains impressively high, surpassing 90%, 97%, and 99% for one-shot, three-shot, and five-shot recognition, respectively. This finding demonstrates the system's robustness even under a limited number of receivers. As a result,

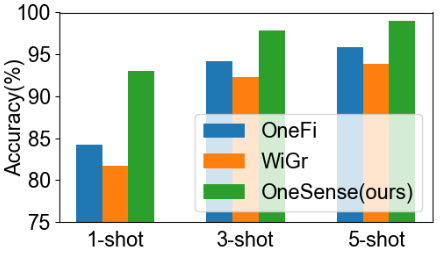


Fig. 7: Comparison with state-of-the-art.

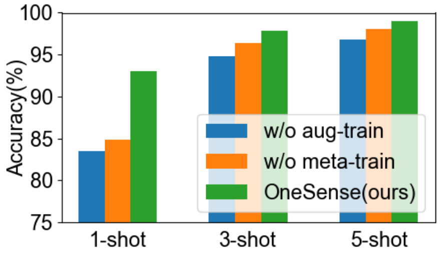


Fig. 8: Effect of AML framework.

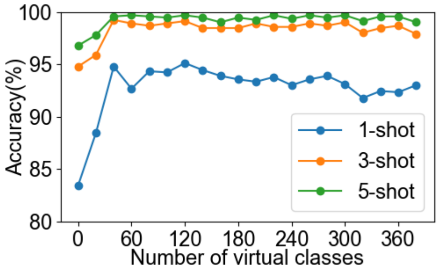


Fig. 9: Effect of data enrichment.

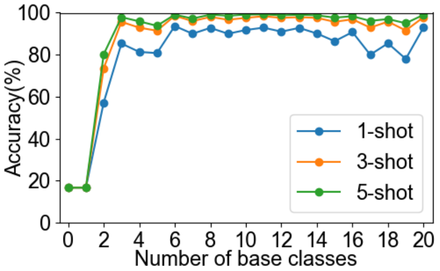


Fig. 10: Effect of no. of base classes.

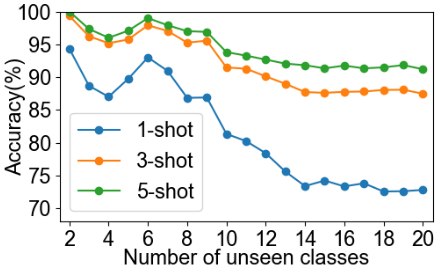


Fig. 11: Effect of no. of unseen classes.

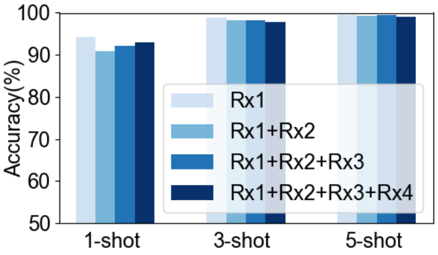


Fig. 12: Effect of no. of receivers.

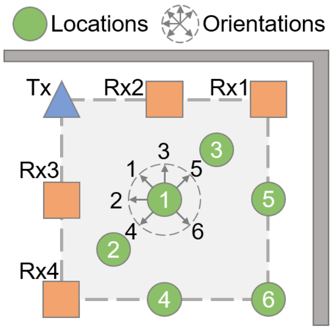


Fig. 13: Different locations and orientations.

OneSense proves to be feasible and effective in resource-constrained environments, while still achieving satisfactory gesture recognition performance.

C. Cross-domain Performance

In this subsection, we evaluate the cross-domain performance of *OneSense*, where the system is pre-trained in one domain while fine-tuned and tested in another. The considered domains include environment, location (the user's position within the sensing area), and orientation (the user's orientation with respect to the transmitter), as shown in Fig. 13.

Cross-environment performance. We mark the environments of the laboratory, dining room, living room, and office as Env1, 2, 3, and 4, respectively. To assess the cross-environment performance, we first pre-train the feature extractor with samples collected in Env1. Subsequently, we fine-tune the classifier and calculate accuracies using data from Env1, 2, 3, and 4, respectively. As depicted in Fig. 14, the accuracies for all these environments are high, which indicates the outstanding cross-environment performance of *OneSense*. This is reasonable as we extract environment-independent Doppler spectrogram to enable gesture recognition. Thus, once *OneSense* was bootstrapped in an environment, it can be deployed in any other environment for accurate gesture recognition.

Cross-location performance. Next, we investigate the effect of user locations on the system's performance. To this end, we first collect data at six locations (marked as Loc1 to Loc6). Then, we pre-train the feature extractor using the data collected at Loc1 and fine-tune the classifier using data collected from all six locations, respectively. Fig. 15 presents the recognition accuracies for one-shot, three-shot, and five-shot scenarios. It can be observed that, although there may be some variances in performance, *OneSense* can achieve high accuracy at most locations, indicating its notable adaptability to different user locations. This capability is essential for real-world scenarios, where users may perform gestures at various locations.

Cross-orientation performance. Similar to the cross-location experiment, to assess cross-orientation performance, we conduct pre-training on data from the first orientation (Ori1), and subsequently fine-tune the classifier using few-shot data from all six orientations (Ori1 to Ori6). As illustrated in Fig. 16, *OneSense* demonstrates commendable performance across most orientations, while encountering some challenges in Ori6. This is primarily because Ori6 involves the user performing gestures with their back toward the transmitter, resulting in significant signal attenuation. Nonetheless, the accuracy achieved at Ori6 surpasses 93% in three-shot settings. Overall, *OneSense* exhibits satisfactory cross-orientation performance, showcasing its robustness in handling different orientations.

D. Time Cost on AML

The time costs of AML mainly come from two components: two-stage pre-training and fine-tuning. In this subsection, we

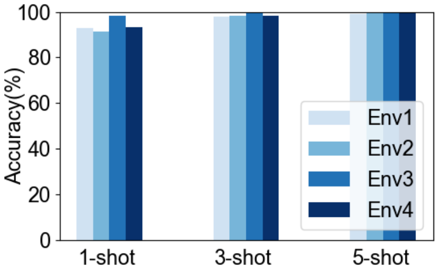


Fig. 14: Cross-environment performance.

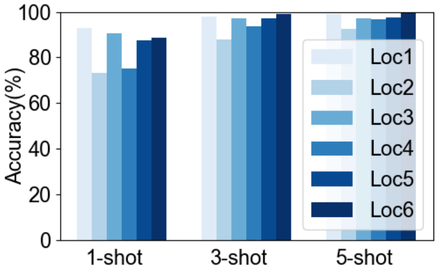


Fig. 15: Cross-location performance.

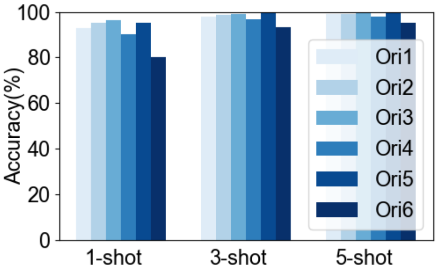


Fig. 16: Cross-orientation performance.

assess the time costs of these two components based on an NVIDIA RTX 3080 GPU.

Pre-training. The experimental results show that, *OneSense* requires only 41.8 seconds for pre-training (including aug-training and meta-training stages), to achieve a one-shot recognition accuracy exceeding 90%. In comparison, a traditional meta-learning approach, MAML [7], demands more than 300 seconds of pre-training to achieve an accuracy of convergence (88.3%). It can be found that the delicate design of our AML framework not only reduces the pre-training latency (86.1%+ reduction), but also improves the recognition performance.

Fine-tuning. In the use of *OneSense*, pre-training is done by the developer, and the user only needs to perform fine-tuning. As only a classifier with a few parameters needs to be updated in fine-tuning, the tuning takes only 2.52 seconds. This means that the gesture can be recognized quickly, demonstrating the outstanding real-time performance of *OneSense*.

VIII. Related Work

Gesture recognition is a critical and active research area, enabling a wide range of applications, such as smart shopping [18] and virtual reality [4]. Traditional gesture recognition approaches often capture the gesture information using cameras [19]–[21], wearable devices [22]–[24], or sonars [25]–[27]. Although demonstrating high recognition accuracy, these approaches have inherent limitations. Camera-based systems only work under good lighting and line-of-sight conditions. Cameras also raise concerns about privacy leakage. Wearable devices offer on-body sensing, but impose user inconvenience. Sonar-based solutions are limited in their sensing range. To address these limitations, WiFi signals are exploited to achieve gesture recognition [4], [28]–[35], as WiFi-based sensing bears several appealing advantages, including visual privacy preservation, robustness to occlusion, and ubiquitous infrastructures [36]–[40].

Existing WiFi-based approaches typically extract features from CSI, and map them to human gestures using learning models. However, most of them require massive training samples to get decent recognition accuracy, which poses significant challenges in data collection and annotation. Recent years have seen some WiFi-based works exploring few-shot gesture recognition, aiming to recognize gestures with only a small number of labeled samples [5], [17], [41], [42]. However, they still face high overhead and inadequate accuracy. For example, OneFi [5] leverages virtual sample generation and transfer learning to achieve few-shot unseen gesture recognition. Yet, its virtual sample generation requires estimating velocity distributions from CSI on at least three receivers, making it time-consuming and resource-intensive. Another solution, WiGr [17], employs a modified prototypical network to improve the recognition performance under few-shot conditions. However, when the number of gestures changes, WiGr needs to re-train the entire model from scratch, resulting in additional computational overhead. Moreover, the one-shot recognition accuracy of these approaches cannot reach 90%, which is somewhat not satisfactory for realistic applications.

To overcome these challenges, we propose *OneSense*, a WiFi-based one-shot gesture recognition system that achieves high accuracy with only one labeled sample per gesture class. *OneSense* addresses the overhead and accuracy issues faced in existing few-shot approaches by sophisticated virtual gesture generation and learning framework designs, which provides a promising solution for real-world deployment.

IX. Conclusion

In order to improve the scalability of WiFi-based gesture recognition to new gesture classes, this paper proposes a novel solution called *OneSense*. In its design, we first present a virtual gesture generation method based on the signal propagation model to enrich the training data. Then, an AML framework is devised to enable scalable one-shot gesture recognition on one hand, and greatly reduce the model training overhead on the other. Extensive real-world experiments show that *OneSense* can achieve 93% one-shot gesture recognition accuracy. Meanwhile, the performance of *OneSense* will not degrade with the changes of the environment, user location, or user orientation.

Acknowledgment

This paper is supported by the National Natural Science Foundation of China under grant U21A20462 and 62372400, Research Institute of Cyberspace Governance in Zhejiang University, “Pioneer” and “Leading Goose” R&D Program of Zhejiang under grant No. 2023C01033, Basic Research Project of Shenzhen Science and Technology Innovation Commission (Project No. JCYJ20190812155213250), Shenzhen Longhua District Science and Technology innovation special fund project (Project No. JCYJ201903).

References

[1] W. Jiang, C. Miao, F. Ma, S. Yao, Y. Wang, Y. Yuan, H. Xue, C. Song, X. Ma, D. Koutsonikolas, W. Xu, and L. Su, "Towards environment independent device free human activity recognition," in *Proceedings of the ACM Annual International Conference on Mobile Computing and Networking (MobiCom)*, 2018.

[2] R. H. Venkatnarayan, G. Page, and M. Shahzad, "Multi-user gesture recognition using wifi," in *Proceedings of the ACM Annual International Conference on Mobile Systems, Applications, and Services (MobiSys)*. ACM, 2018, pp. 401–413.

[3] A. Virmani and M. Shahzad, "Position and orientation agnostic gesture recognition using wifi," in *Proceedings of the ACM Annual International Conference on Mobile Systems, Applications, and Services (MobiSys)*, 2017.

[4] Y. Zheng, Y. Zhang, K. Qian, G. Zhang, Y. Liu, C. Wu, and Z. Yang, "Zero-effort cross-domain gesture recognition with wi-fi," in *Proceedings of the ACM Annual International Conference on Mobile Systems, Applications, and Services (MobiSys)*, 2019.

[5] R. Xiao, J. Liu, J. Han, and K. Ren, "Onefi: One-shot recognition for unseen gesture via COTS wifi," in *Proceedings of the ACM Conference on Embedded Networked Sensor Systems (SenSys)*, 2021.

[6] O. Vinyals, C. Blundell, T. Lillicrap, K. Kavukcuoglu, and D. Wierstra, "Matching networks for one shot learning," in *Proceedings of the Annual Conference on Neural Information Processing Systems (NIPS)*, 2016.

[7] C. Finn, P. Abbeel, and S. Levine, "Model-agnostic meta-learning for fast adaptation of deep networks," in *Proceedings of the International Conference on Machine Learning (ICML)*, 2017.

[8] J. Snell, K. Swersky, and R. S. Zemel, "Prototypical networks for few-shot learning," in *Proceedings of the Annual Conference on Neural Information Processing Systems (NIPS)*, 2017.

[9] Y. Ma, G. Zhou, and S. Wang, "Wifi sensing with channel state information: A survey," *ACM Computing Surveys*, vol. 52, no. 3, pp. 46:1–46:36, 2019.

[10] D. Tse and P. Viswanath, *Fundamentals of Wireless Communication*. Cambridge University Press, 2005.

[11] R. Vilalta and Y. Drissi, "A perspective view and survey of meta-learning," *Artificial Intelligence Review*, vol. 18, no. 2, pp. 77–95, 2002.

[12] K. Qian, C. Wu, Z. Zhou, Y. Zheng, Z. Yang, and Y. Liu, "Inferring motion direction using commodity wi-fi for interactive exergames," in *Proceedings of the Annual ACM Conference on Human Factors in Computing Systems (CHI)*, 2017.

[13] W. Wang, A. X. Liu, M. Shahzad, K. Ling, and S. Lu, "Understanding and modeling of wifi signal based human activity recognition," in *Proceedings of the ACM Annual International Conference on Mobile Computing and Networking (MobiCom)*, 2015.

[14] K. Niu, X. Wang, F. Zhang, R. Zheng, Z. Yao, and D. Zhang, "Rethinking doppler effect for accurate velocity estimation with commodity wifi devices," *IEEE Journal on Selected Areas in Communications (JSAC)*, vol. 40, no. 7, pp. 2164–2178, 2022.

[15] J. Wang and J. R. Jang, "Training a singing transcription model using connectionist temporal classification loss and cross-entropy loss," *IEEE ACM Transactions on Audio Speech and Language Processing*, vol. 31, pp. 383–396, 2023.

[16] D. Halperin, W. Hu, A. Sheth, and D. Wetherall, "Tool release: gathering 802.11n traces with channel state information," *Computer Communication Review (CCR)*, vol. 41, no. 1, p. 53, 2011.

[17] X. Zhang, C. Tang, K. Yin, and Q. Ni, "Wifi-based cross-domain gesture recognition via modified prototypical networks," *IEEE Internet of Things Journal (IoTJ)*, vol. 9, no. 11, pp. 8584–8596, 2022.

[18] K. Cui, Y. Wang, Y. Zheng, and J. Han, "Shakereader: 'read' UHF RFID using smartphone," *IEEE Transactions on Mobile Computing (TMC)*, vol. 22, no. 3, pp. 1793–1809, 2023.

[19] G. Gkioxari, R. B. Girshick, P. Dollár, and K. He, "Detecting and recognizing human-object interactions," in *Proceddings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[20] T. Li, Q. Liu, and X. Zhou, "Practical human sensing in the light," in *Proceedings of the ACM Annual International Conference on Mobile Systems, Applications, and Services (MobiSys)*, 2016.

[21] M. Wang, B. Ni, and X. Yang, "Recurrent modeling of interaction context for collective activity recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[22] A. Bulling, U. Blanke, and B. Schiele, "A tutorial on human activity recognition using body-worn inertial sensors," *ACM Computing Surveys*, vol. 46, no. 3, pp. 33:1–33:33, 2014.

[23] A. I. Withana, R. L. Peiris, N. Samarasekara, and S. Nanayakkara, "zsense: Enabling shallow depth gesture recognition for greater input expressivity on smart wearables," in *Proceedings of the Annual ACM Conference on Human Factors in Computing Systems (CHI)*. ACM, 2015, pp. 3661–3670.

[24] T. Zhao, J. Liu, Y. Wang, H. Liu, and Y. Chen, "Ppg-based finger-level gesture recognition leveraging wearables," in *Proceedings of the IEEE Conference on Computer Communications (INFOCOM)*, 2018.

[25] K. Kalgaonkar and B. Raj, "One-handed gesture recognition using ultrasonic doppler sonar," in *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, 2009.

[26] K. Yatani and K. N. Truong, "Bodyscope: a wearable acoustic sensor for activity recognition," in *Proceedings of the ACM Conference on Ubiquitous Computing (Ubicomp)*, 2012.

[27] R. Nandakumar, A. Takakuwa, T. Kohno, and S. Gollakota, "Covertband: Activity information leakage using music," *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies (IMWUT)*, vol. 1, no. 3, pp. 87:1–87:24, 2017.

[28] Q. Pu, S. Gupta, S. Gollakota, and S. N. Patel, "Whole-home gesture recognition using wireless signals," in *Proceedings of the ACM Annual International Conference on Mobile Computing and Networking (MobiCom)*, 2013.

[29] P. Melgarejo, X. Zhang, P. Ramanathan, and D. Chu, "Leveraging directional antenna capabilities for fine-grained gesture recognition," in *Proceedings of the ACM Conference on Ubiquitous Computing (UbiComp)*, 2014.

[30] H. Abdelnasser, M. Youssef, and K. A. Harras, "Wigest: A ubiquitous wifi-based gesture recognition system," in *Proceedings of the IEEE Conference on Computer Communications (INFOCOM)*, 2015.

[31] S. Ji, Y. Xie, and M. Li, "Sifall: Practical online fall detection with RF sensing," in *Proceedings of the ACM Conference on Embedded Networked Sensor Systems (SenSys)*, 2022.

[32] F. Adib and D. Katabi, "See through walls with wifi!" in *Proceedings of the ACM Conference of the Special Interest Group on Data Communication (SIGCOMM)*, 2013.

[33] Y. Wang, K. Wu, and L. M. Ni, "Wifall: Device-free fall detection by wireless networks," *IEEE Transactions on Mobile Computing (TMC)*, vol. 16, no. 2, pp. 581–594, 2017.

[34] S. Palipana, D. Rojas, P. Agrawal, and D. Pesch, "Falldefi: Ubiquitous fall detection using commodity wi-fi devices," *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies (IMWUT)*, vol. 1, no. 4, pp. 155:1–155:25, 2017.

[35] D. Zhang, H. Wang, Y. Wang, and J. Ma, "Anti-fall: A non-intrusive and real-time fall detector leveraging CSI from commodity wifi devices," in *Proceedings of the Conference on Smart Homes and Health Telematics (ICOST)*, 2015.

[36] Y. Xu, W. Yang, J. Wang, X. Zhou, H. Li, and L. Huang, "Wistep: Device-free step counting with wifi signals," *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies (IMWUT)*, vol. 1, no. 4, pp. 172:1–172:23, 2017.

[37] F. Wang, J. Han, F. Lin, and K. Ren, "Wipin: Operation-free passive person identification using wi-fi signals," in *Proceedings of the IEEE Global Communications Conference (GLOBECOM)*, 2019.

[38] X. Liu, J. Cao, S. Tang, J. Wen, and P. Guo, "Contactless respiration monitoring via off-the-shelf wifi devices," *IEEE Transactions on Mobile Computing (TMC)*, vol. 15, no. 10, pp. 2466–2479, 2016.

[39] L. Gong, W. Yang, D. Man, G. Dong, M. Yu, and J. Lv, "Wifi-based real-time calibration-free passive human motion detection," *Sensors*, vol. 15, no. 12, pp. 32 213–32 229, 2015.

[40] Y. Gu, J. Zhan, Y. Ji, J. Li, F. Ren, and S. Gao, "Mosense: An rf-based motion detection system via off-the-shelf wifi devices," *IEEE Internet of Things Journal (IoTJ)*, vol. 4, no. 6, pp. 2326–2341, 2017.

[41] J. Yang, H. Zou, Y. Zhou, and L. Xie, "Learning gestures from wifi: A siamese recurrent convolutional architecture," *IEEE Internet of Things Journal (IoTJ)*, vol. 6, no. 6, pp. 10 763–10 772, 2019.

[42] X. Ding, T. Jiang, Y. Zhong, Y. Huang, and Z. Li, "Wi-fi-based location-independent human activity recognition via meta learning," *Sensors*, vol. 21, no. 8, p. 2654, 2021.