CrossSense: Towards Cross-Site and Large-Scale WiFi Sensing

Jie Zhang, Zhanyong Tang*, Meng Li, Dingyi Fang
Northwest University, China

Petteri Nurmi
Lancaster University, U. K.
University of Helsinki, Finland

Zheng Wang*
Lancaster University, U. K.

ABSTRACT

We present CrossSense, a novel system for scaling up WiFi sensing to new environments and larger problems. To reduce the cost of sensing model training data collection, CrossSense employs machine learning to train, off-line, a *roaming model* that generates from one set of measurements synthetic training samples for each target environment. To scale up to a larger problem size, CrossSense adopts a *mixture-of-experts* approach where multiple specialized sensing models, or *experts*, are used to capture the mapping from diverse WiFi inputs to the desired outputs. The experts are trained offline and at runtime the appropriate expert for a given input is automatically chosen. We evaluate CrossSense by applying it to two representative WiFi sensing applications, gait identification and gesture recognition, in controlled single-link environments. We show that CrossSense boosts the accuracy of state-of-the-art WiFi sensing techniques from 20% to over 80% and 90% for gait identification and gesture recognition respectively, delivering consistently good performance – particularly when the problem size is significantly greater than that current approaches can effectively handle.

CCS CONCEPTS

• **Networks** → *Wireless access networks*; • **Human-centered computing** → *Ubiquitous and mobile computing systems and tools*;

KEYWORDS

Wireless sensing; Channel state information; Received signal strength indicator; Machine learning

ACM Reference Format:
Jie Zhang, Zhanyong Tang*, Meng Li, Dingyi Fang, Petteri Nurmi, and Zheng Wang. 2018. CrossSense: Towards Cross-Site and Large-Scale WiFi Sensing. In *The 24th Annual International Conference on Mobile Computing and Networking (MobiCom '18), October 29-November 2, 2018, New Delhi, India.* ACM, New York, NY, USA, 16 pages. https://doi.org/10.1145/3241539.3241570

*Corresponding faculty authors: Zhanyong Tang (zytang@nwu.edu.cn) and Zheng Wang (z.wang@lancaster.ac.uk)

MobiCom '18, October 29-November 2, 2018, New Delhi, India

ACM ISBN 978-1-4503-5903-0/18/10…$15.00
https://doi.org/10.1145/3241539.3241570

1 INTRODUCTION

WiFi has emerged as a powerful medium for sensing information. By measuring how the wireless channel is affected by humans and their activities, tasks like gait identification [76], gesture recognition [2, 26], activity recognition [59, 75] and even vital sign monitoring [1, 68] could be possible. WiFi sensing is particularly attractive for smart spaces as it can be easily deployed using as few as two wireless routers, not requiring instrumenting the users, and being less privacy intrusive than other infrastructure-based solutions such as video monitoring [28, 33].

While existing research has demonstrated the vast potential of WiFi as a sensing technique, currently there are two significant drawbacks that limit the uptake of WiFi sensing and the scale at which it can operate. Firstly, the dominant approach for WiFi sensing requires a labour-intensive and time-consuming process of collecting training measurements or fingerprints to characterize how wireless channel metrics, such as channel state information (`CSI`) or received signal strength indicator (`RSSI`), are affected by the target (*e.g., a gait or a gesture*). These training measurements must be carefully collected for each target subject or activity from each deployment site. While it may be feasible to collect such data from each occupant of a home, asking each employee or visitor to provide training measurements from each meeting room in a smart office setting is infeasible. Secondly, many WiFi sensing solutions currently can handle only small sets of subjects. As shown in the paper, performance of prior WiFi sensing techniques decreases quickly as the number of targets goes beyond a handful. This makes deployments into

large-scale settings, such as enterprises or campuses, infeasible as these scenarios would require supporting hundreds of users and numerous activities.

In this paper, we present CrossSense, a system for scaling up WiFi sensing to new environments and larger sensing problems. By addressing both challenges simultaneously, CrossSense not only increases the scale and uptake of WiFi sensing, but opens up new possibilities for innovative applications and services. As an example, one can offer personalized services in *every* room within a company building, hotel or conference center by collecting a training sample from the visitor *once* at the reception; and the system can now support many more users at a time. WiFi sensing could also be part of a multi-factor authentication protocol [29, 35, 53, 65] to verify a user's claimed identity across sites. These exciting applications can only be realized if the WiFi training measurements can be effectively translated and utilized across sites, and if sensing can target a larger problem size. CrossSense has been designed to offer these capabilities[1].

To enable WiFi training measurements to be collected once and used across sites, CrossSense integrates an artificial neural network based approach for roaming previously collected measurements from another site to automatically generate synthetic sensing model training samples (also known as virtual samples [66]) for the new environment. The roaming model is first trained *off-line* and then used for any *unseen* subjects. To reduce the number of samples required for learning a roaming model for a new site or sensing task, we employ *transfer learning* [87] to leverage an existing roaming model. We exploit the fact that the WiFi signal properties that are abstracted by the beginning layers of the neural network are mostly independent of the optimization problem. We reuse these parts of the network across sites and domains, and, in the process, we speed up learning new roaming models considerably.

To enable WiFi sensing to target a larger problem size, we employ a *mixture-of-experts* based approach [30]. The central idea is that, instead of using a single monolithic model for a sensing task (e.g., gait recognition or gesture recognition), we use multiple models (*experts*) where each expert is specialized for a subset of application scenarios and input data. In this way, each model is used only when its predictions are effective. One major advantage of our approach is that new models can easily be added and are selected only when appropriate. The result is a new way of using machine learning for WiFi sensing, with a generalized framework for diverse inputs, application domains and environments.

We demonstrate the benefits of CrossSense by applying it to WiFi-based gait identification and gesture recognition, and considering `CSI` and `RSSI` based metrics. We compare to five state-of-the-art WiFi sensing methods [26, 60, 66, 76, 90] and a wireless signal translation approach [13]. We perform an extensive evaluation across three controlled sites, involving 100 users, 40 gestures, and over 1.2 million wireless activity samples. We show that CrossSense delivers *the best and the most reliable* performance across evaluation scenarios and sensing tasks. It boosts the accuracy of state-of-the-art solutions by 4 folds, and enables WiFi sensing to work across sites and for significantly larger problem sizes.

The main contributions of this paper are:

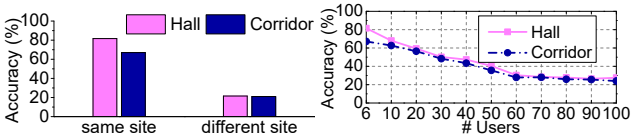
- We present an automatic scheme to effectively leverage existing WiFi training measurements to build sensing models for new environments (Section 4);
- We apply, for the first time, transfer learning to effectively reuse the learned knowledge across different sites and tasks for WiFi sensing (Section 4.3);
- Our work is the first to employ mixture-of-experts for WiFi sensing to scale up the size of the problems that can be supported, delivering significant performance improvement over state-of-the-art techniques. Our generic framework allows new models to be easily added to target a wider range of application contexts (Section 5);

2 MOTIVATION

CrossSense addresses two significant drawbacks in current wireless sensing solutions, cross-site generalization and scale of the problems that can be supported. To further illustrate these limitations, in the following we consider gait identification as a representative example of WiFi sensing tasks and demonstrate how the performance of a state-of-the-art method, WiWho [90], suffers when (a) the sensing model is built from training samples collected from *another* environment, and (b) targeting *more than a dozen of users*.

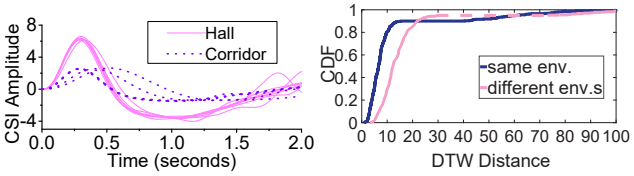
Setup. Our evaluation environments include a hall and a corridor. The two environments are of different sizes and thus have different multipath effects on the wireless signal. More details of our evaluation settings are given in Section 6.1. To collect training measurements, each user walked on a straight line and passed two WiFi devices 20 times. During testing, each user walked pass the evaluation scene on a straight line 10 times. The task of WiWho is to map a testing measurement to one of the known users.

[1]Note that our work specifically targets WiFi sensing techniques that rely on a classifier-based approach to distinguish between target activities. While there have been some efforts at WiFi sensing techniques that model target activities directly through signal characteristics, most notably in gesture recognition where gestures can be modelled as a sequence of primitives (such as moving the hand left or right) that are captured from Doppler shift information [2, 32, 50], the majority of past techniques in WiFi sensing are based on classifier-based techniques.



(a) (b)

Figure 1: The accuracy of WiWho drops significantly if it is not tuned for the target environment (a), and if the number of target users is more than a dozen (b).



(a) (b)

Figure 2: The CSI patterns for the same person from two different environments are different.

Impact of training data. The first experiment involves six users – the same number of users used to evaluate WiWho in [90]. We expect WiWho to work effectively on this user size. Figure 1a shows that a high accuracy (75% to 87%) can be achieved if WiWho is built using examples obtained from the testing environment. The performance is comparable to the accuracy of 75% to 80% reported in [90]. However, the accuracy drops to below 25% when the sensing model is built using samples collected from another site.

Environmental impact. Figure 2a shows that the CSI amplitudes collected for the same person is visually different across sites. To quantify the differences, we calculate the dynamic time warping (DTW) distance between each pair of the 10 measurements collected for this person. This metric is often used to quantify the similarity between signal measurements [69, 73] – the further the DTW distance is, the less similar two measurements are. The cumulative distribution function (CDF) diagram in Figure 2b suggests that a larger number of samples have a closer DTW distance when the measurements are taken from the same environment over those taken from different environments. The average DTW distance is increased by 40% when the wireless measurements are taken from a different environment, indicating that the testing environment has a great impact on wireless channel metrics. We also found that using directional antennas does not eliminate the problem, which is in line with the finding in prior research [66]. Clearly, a sensing model tuned for the hall will be ineffective for the corridor.

Impact of problem sizes. In the second experiment, we collect the training samples from the target environment, but we consider more target users. Figure 1b shows that the accuracy drops quickly as the number of users increases. The accuracy drops from around 80% to below 60% and 30% when targeting more than 10 and 60 users, respectively. Later in this paper, we demonstrate that the same issue is observed for gesture recognition – as the number of gestures increases, the accuracy of current sensing methods decreases greatly.

Lessons learned. This example shows that current classifier-based approaches have to collect the training samples of each target subject or activity from each deployment environment and only work for a small problem size. We want to have a technique that can utilize the WiFi training samples collected in one site for each new environment. If we can achieve this, we can then build sensing models for new environments using the same set of training samples at a low cost. We would also like the sensing method to work effectively on a larger problem size to support more subjects and activities. CrossSense has been designed to address both issues.

3 OVERVIEW OF CROSSSENSE

The focus of our work is on enabling WiFi sensing (1) to work efficiently across sites by using a single set of training measurements, and (2) to scale to large problem sizes.

To enable cross-site sensing, CrossSense employs machine learning to learn a function to *roam* the WiFi training measurements collected from one environment to another. The roaming model takes in WiFi measurements collected from the data collection site and generates synthetic training instances as if the synthesize data were collected from the target environment. The model is trained *off-line* on a set of examples collected from the data collection site and each deployment site. The learned model is used to translate training measurements for any *unseen* subjects or activities. Crucially, the roaming model needs to capture how the environment affects the WiFi signal from a small set of examples, so that we can apply the model to synthesize a larger set of *unseen* measurements. This is detailed at Section 4.

To scale WiFi sensing to larger problem sizes, CrossSense uses multiple models to map the *on-site* WiFi measurement to the desired output. It then uses a classifier to select which expert should be used for the input measurement. Depending on the characteristics of the input signal, different models can be dynamically selected. This is different from all prior WiFi sensing studies that employ a *one-size-fits-all* model. We argue that a single model is unlikely to precisely capture the diverse inputs for a large problem size. This is described in more details at Section 5.

4 ROAMING WIFI MEASUREMENTS

CrossSense uses an artificial neural network (ANN) for translating WiFi sensing measurements across environments. The motivation for using ANN is twofold. Firstly, ANN can capture both linear and non-linear relationships, making it well

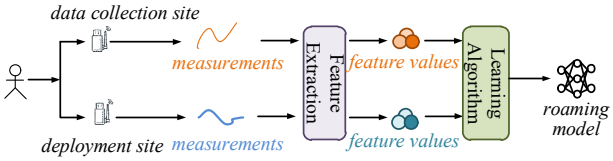


Figure 3: Training the roaming model.

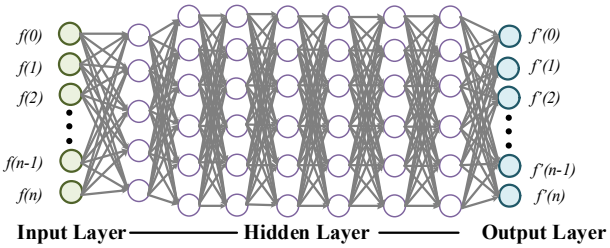


Figure 4: Our roaming model is a multi-layer neural network. It takes a WiFi measurement and produces a synthetic sample for the target environment.

suited for generalizing across diverse environments. For example, open halls tend to have different multipath effects than small corridors, requiring a flexible way to model diverse types of relationships. Secondly, ANNs are well-suited for transfer learning and can be used to reduce training costs for new environments (Section 4.3). In Section 7.7, we also consider other techniques and conclude that ANNs have the best overall performance.

The inputs to the roaming model are numerical feature vectors extracted from previously collected WiFi training measurements, and the outputs are the *expected* feature values of those measurements in the target environment. The set of features used is described in Section 5.1. Building and using such a model follows the 3-step process for supervised machine learning: (i) generate training data (ii) learn a model (iii) use the model. These steps are described as follows.

4.1 Building a Roaming Model

Figure 3 illustrates the process for building a roaming model. We first collect some WiFi signal measurements from the deployment and the data collection sites. We then extract important quantifiable properties, or *features*, of the raw signal data. Finally, we apply a learning algorithm over the collected data to construct the roaming model.

Figure 4 depicts the structure of our roaming model, which is a fully connected, feed-forward ANN with 7 hidden layers. The number of nodes of the input and the output layers is determined by the dimensionality of the feature set (see Table 5.1). The trained, fully-connected network can be pruned to remove some of the network pathways [31]. We stress that keeping the network structure simple is essential for learning an effective model from a relatively small training dataset. We also evaluate various neural network structures. This is discussed in Section 7.7.

4.1.1 Generate Roaming Model Training Data. In this work, roaming model training data are collected from some users. The subjects (e.g., gaits or gestures) involved in training do not need to be the same as the ones to be targeted during deployment. This is because the goal is to learn the wireless representation translation for a particular task (e.g., gait identification or gesture recognition) rather than for a specific subject. Specifically, the roaming model learns from training samples how to map a WiFi measurement from one environment to another. Once the model has learned the mapping, it can then apply the learned knowledge to measurements of *unseen* subjects. In our experiments, the training dataset consists of gait and gesture data collected from the data collection site and each deployment site. A training user needs to perform the same activity (walking or performing a gesture) in both the deployment and the training environments. The initial training involves 50 and 10 users respectively to learn a roaming model for gait identification and gesture recognition for *the first* deployment site. Using *transfer learning* [87], the number of roaming model training samples can be reduced by 4x when targeting additional sites without sacrificing performance (see Section 7.5).

Noise control. Unlike past work that asks users to repeat an activity a fixed number of times, we use statistical reasoning to determine the times on a per user basis. Specifically, we calculate the confidence range of the measurements based on a 95% confidence interval for four metrics: the maximum, average, minimum, and standard deviation values of each user's measurements; we make sure that the confidence range of each metric is within 10% of its mean. As a result, in our experiments, a user on average needs to walk 10 times and repeat a gesture 5 times to gather clean data. Walking pass wireless receivers and performing a gesture take less than 5 and 2 seconds respectively.

Training instance pairing. Recall that we need to learn, from the roaming model training data, how to map a subject's WiFi measurement from one site to the other. The mapping relation must be captured by the training examples. As such, a roaming model training instance consists of measurements gathered from the data collection site and the deployment site for a training subject or activity. Because we have taken multiple measurements from both sites, a sample from one site can be mapped to any of the samples gathered from the other site for the same subject (e.g., a user or gesture). To determine the mapping, we use a linear regression model to fit the training data. Specifically, we enumerate all possible mappings and then choose a mapping that leads to a regression fitting with the smallest mean squared error. Furthermore, instead of presenting the raw measurement values to the

learning algorithm, we transform the WiFi signal data into a fixed length feature vector. As detailed in Section 5, we use a range of features to capture diverse behaviors. Therefore, we generate, from the raw signal measurements of each site, a training dataset for each feature set.

4.1.2 Learning a Roaming Model. After gathering sufficient training data, we build the roaming model by employing a supervised learning algorithm. The algorithm takes in feature values of the data gathered from (1) the data collection site (*input*) and (2) the deployment site (*output*). It produces a roaming model for each feature set. Training is a *one-off* cost performed off-line, and the learned model can be used for any *new*, *unseen* subjects without incurring extra training. Therefore, the training cost can be amortized when the number of activities to be targeted is greater than that involved in training the roaming model, which is likely to be the case for large-scale deployments.

Our roaming model is trained using Back-propagation with Stochastic Gradient Descent (SGD). For a set of training examples $X_1 \ldots X_n$, SGD attempts to find the network parameters Θ that minimize the output of a loss function:

$$\Theta = \arg\min_{\Theta} \frac{1}{n} \sum_{i=1}^{n} \ell\left(X_i, \Theta\right)$$

where loss function $\ell\left(x, \Theta\right)$ computes the logarithmic difference between the model output and expected values. Intuitively, the goal of the loss function is to make sure that samples for the same subject are as similar as possible while samples from different activities are sufficiently discriminative. It is to note that we use grid-based search to automatically find the optimal hyper-parameters to minimize the loss on our training data.

Training cost. The total training time is comprised of three parts: gathering the raw wireless channel metrics, processing the raw data, and then building a model. Gathering the raw data consumes most of the total training time, in this work it took about 14 hours to collect gait and gesture data from 50 users from one environment. In comparison processing the raw data and learning the model took a negligible amount of time, less than 20 minutes on a multi-core server.

4.2 Deployment

Once we have built and trained the roaming models – one model per site, per feature set, we can use them to quickly translate the sensing model training measurements for each target subject from one site to another. An alternative is to map measurements from the deployment environment to the training environment, and then use the same classifier for activity sensing. There is no difference in terms of runtime overhead for this alternative and our approach. Exploiting this alternative is part of our future work.

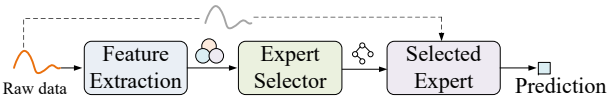


Figure 5: Overview of our mixture-of-experts approach.

4.3 Transfer Learning

Prior work in machine learning has shown that ANN models trained on similar inputs for different tasks often share useful commonalities. The idea is that the information learned at the early layers of a neural network (i.e., those closer to the input layer) are used to capture the input and are mostly independent of the optimization goal. The later the network layers are, the more specialized the layers become.

Transfer learning has been demonstrated to be effective in processing mobile sensor data [72, 95], and we believe it can be applied to wireless signals too. That is, we can utilize e.g., a trained roaming model between sites A and B to speed up the process for learning a model between sites A and C. This is because the first few layers of our ANN model are likely to focus on abstracting the input WiFi channel metrics and are largely independent to the model output. Since we use the same network structure, transfer learning is as simple as copying the learned weights of the learned model to initialize the new network. Then we train the model as normal but with fewer training instances. In Section 7.5, we show that transfer learning greatly reduces the training cost across deployment sites and tasks (e.g., using a gait translation model to speed up training of gesture translation).

5 SCALING TO LARGE PROBLEM SIZES

CrossSense uses multiple distinct expert models to scale WiFi sensing to larger problem sizes. The expert models are built from the WiFi training measurements of the target subjects such as users or gestures. When targeting a new environment, we use the *synthetic* training samples produced by our roaming model to tune the expert models for the new environment. During deployment, an expert selector decides which model to use, based on the on-site signal measurement. This process is illustrated at Figure 5.

5.1 Wireless Signal Features

Table 1 gives the set of features considered in this work. We use a range of features that were found to be useful in prior work of WiFi sensing, but new features can be added. These features are grouped into four categories: (1) statistics of the wireless channel [2, 60, 90]; (2) compression features [9, 34] for which we apply the Discrete Wavelet Transform [67] to the wireless channel metrics to sample the time and frequency information; (3) spectrogram features such as the normalized energy at the main frequency [76]; and (4) transformed features for which we covert the raw wireless channel measurements to a different form where discriminative

Table 1: Wireless signal features used in this work

Statistical Features
Time domain: min/max, min/max 10th/90th, mean, std. deviation, etc. over a time window;
Frequency domain: energy, domain-frequency ratio, FFT peaks;
Compression Features
3-level discrete wavelet transform on the signal data;
Spectrogram Features
normalized energy at the main frequency, the maximum, minimum, average, and variance for the movement speed, etc.
Transformed Features
The signal shape; the autocorrelation and the partial autocorrelation functions, etc.

Table 2: Classification techniques used in the work

Naive Bayes (NB)	Random Forest (RF)
Support Vector Machine (SVM) w/ the RBF kernel	SVM with a linear kernel
K-Nearest Neighbor (KNN)	Adaboost

features can be easily found [12]. To extract the features, we first remove noise from the raw signal measurement (see Section 5.3). Next, we extract the feature values from multiple subcarriers of the wireless signal and store the feature values in a fixed vector of real values.

Supervised learning typically requires feature values to lie in a certain range. Therefore, we normalize the value of each features to the range between 0 and 1. We record the maximum and minimum value of each feature found at the training phase, and use these values to scale features extracted from a new signal measurement during deployment.

5.2 Expert Models

To construct the expert models, we consider six classification techniques listed in Table 2. For this work, a classifier takes in a feature vector of the input measurement to predict the target's label that the input corresponds to. Our classifiers are a mixture of linear (e.g., KNN) and non-linear (e.g., SVM with the RBF kernel) models which were proven to be useful in prior wireless sensing tasks. It is worth mentioning that other classifiers can be added and the process of expert model training and selection can remain unchanged.

Each classifier can be used together with one of the feature sets given in Table 1. To decide which feature-model combinations should be used as an expert, we perform cross-validation on the *translated* training measurements. We break the dataset into two parts. We train a feature-model combination on one dataset (*expert model training* set), and test the trained model on another dataset (*expert model testing* set). We keep a combination if it either gives the highest accuracy or can correctly classify testing cases where others fail. As a result, we keep the 10 combinations given in Table 3.

Table 3: Chosen expert models, sorted by capabilities.

ID	Expert: (Feature, Model)	ID	Expert: (Feature, Model)
1	(Transformed, SVM-Linear)	2	(Spectrogram, SVM-RBF)
3	(Statistics, RF)	4	(Transformed, RF)
5	(Transformed, SVM-RBF)	6	(Statistics, SVM-RBF)
7	(Statistics, NB)	8	(Transformed, Adaboost)
9	(Compression, SVM-RBF)	10	(Compression, KNN)

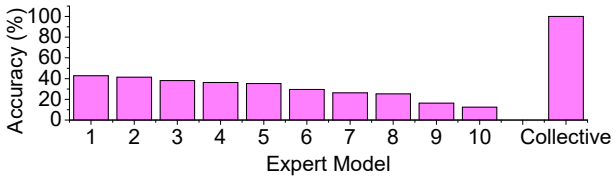


Figure 6: Percentages of samples that to be correctly classified by an expert on a testing dataset.

Expert models are tuned for each environment using the *translated* WiFi measurements produced by our roaming model. Learning in this context is to find model parameters that can closely correlate the feature values of the *translated* measurements to the corresponding subject or activity.

Figure 6 shows the accuracy of each selected expert model on the *expert model testing set* for gait identification. Here we train each expert model on one set of data (i.e., users or gestures) and test it on another set of data. The collective scheme is considered to give the correct answer if any of the expert model in the collection gives the correct prediction. While no individual expert can correctly classify over 50% of the test cases, collectively, they can successfully identify all. Later in Section 7.1, we evaluate this approach on the full dataset and show that the same observation also applies to gesture recognition.

We note that expert models are automatically learned from training data, treating the problem domain and data patterns as black boxes; new applications and data patterns would be learned similarly, potentially causing the addition of new models. Like the roaming model, expert training and selection are also performed *off-line* and are an one-off cost.

5.3 Expert Selector

After learning individual expert models, we need to have a mechanism to determine which expert to use for a given on-site WiFi measurement. Our expert selector is a KNN classifier where k is set to 3 which is determined by performing cross-validation on our training data. We use the *translated* WiFi training measurements of the target subjects or activities as *fingerprints* for choosing expert models. To determine which expert models to use, we find which of the three fingerprints are closest to the input collected during deployment. We then use the experts that are found, during training, to be effective for the closest neighboring fingerprints. Neighbor evaluation is performed by applying the DTW algorithm [52]

to the de-noised wireless channel metrics. We choose to use DTW because it is a well-established method for measuring the similarity between two temporal sequences [73]. We also note that no training is required for KNN because the algorithm works directly on the training samples.

We use a Principal Component Analysis (PCA) based method described in [75] to reduce noise and dimensionality of the raw measurement. We then choose the first 30 principal components given by PCA, which account for over 95% of the data variance. We record the PCA coefficients and use them to transform the input collected during deployment. The KNN algorithm then works on the de-noised input.

Modern WiFi devices typically have multiple transmit and receive antennas and a wireless channel is typically divided into multiple subcarriers. Therefore, the wireless channel metric can be measured independently from each subcarrier. For this reason, we first use KNN to select three experts from each subcarrier measurements, and then use majority voting to choose the final expert model.

Intuitively, learning an expert selector is easier than learning a one-size-fits-all classifier. This is because there are less classification labels to learn – as the number of expert models is typically smaller than the target problem size. It is also because there are often multiple experts that can correctly classify a given input and the expert selector only needs to pick one of them. We also note that one advantage of using KNN is that the distance used for expert selection can also be used to measure the prediction confidence, which essentially provides a degree of soundness guarantee. For example, if an on-site measurement is too far from any fingerprint, this might mean that either we do not have a fingerprint for the target, or the environment has changed significantly where a retraining of the roaming model is required.

5.4 Computational Cost

The computational overhead of mixture-of-experts mainly comes from training the expert models. Expert model training is a one-off cost unless new subjects or activities are targeted. It takes less than 15 minutes to train in parallel our 10 expert models on a multi-core server. This can be further accelerated using multiple servers to provide a near real-time model update. The overhead of selecting and using the expert is negligible, as only one model is used at a time. In our case, the expert selector takes less than 1 second to run, and classification takes less than 0.5 second by running the most expensive expert model, Adaboost, on a PC.

6 EXPERIMENTAL SETUP

We thoroughly evaluate CrossSense using over one million wireless measurements collected from 100 users across three sites. All our experiments were approved by our IRB.

6.1 Evaluation Scenarios and Setup

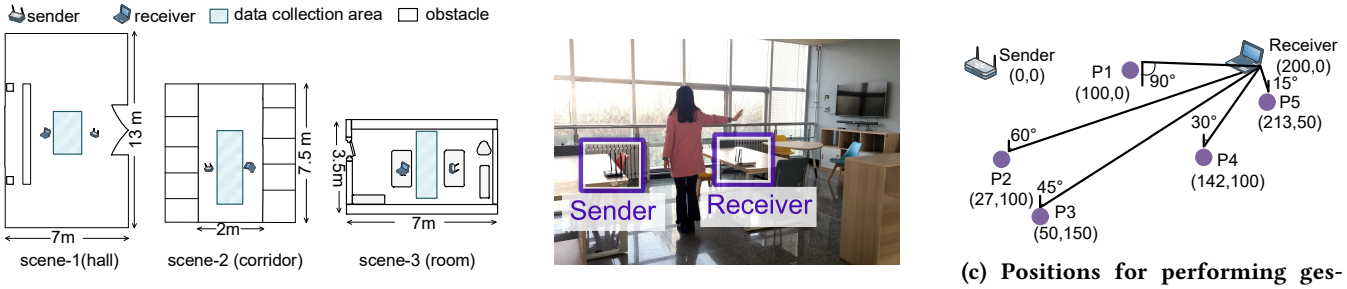
Application scenarios. CrossSense is evaluated on two WiFi sensing applications: gait identification and gesture recognition, and two wireless channel metrics: CSI and RSSI.

Evaluation environments. We test CrossSense in three indoor environments of different sizes. Figure 7a shows the layouts and wireless setups of the three scenes. The first is a spacious hall entrance which mimics the reception area of a building. The second is a smaller, narrow corridor, and the third is a typical indoor environment with furniture include desks, chairs, bookshelves and appliances (see Figure 7b). The CDF diagram in Figure 8a suggests that the multipath effects of our evaluation environments can have a great impact on the wireless channel metrics. A larger number of gait samples (for the same person) have a further distance if the measurements are collected from two different environments. Figure 8b projects 60 measurements (20 per site) for the user seen in Figure 2a onto a 2-dimensional space using PCA. As can be seen from the diagram, measurements from the same environment are more similar to each other; and as a result, we can group measurements into three clusters based on where the data come from.

Wireless setup. As an example, Figure 7b depicts the wireless setup of scene-3. To collect CSI, we used a mini PC with an Intel 5300 NIC as the receiver and a TP-Link WDR7500 wireless router as the transmitter (sender). The sender and the receiver are equipped with three antennas to acquire and record wireless channel measurements from 30 channels for each antenna. We run an open source CSI measurement tool [24] on the mini PC to obtain CSI measurements in a 5GHz WiFi environment. To measure the CSI, the sender pings the receiver at a rate of 1,000 packets per second, which is a standard sample rate used in past work [51, 85]. We use a XiaoMI note2 smartphone as the receiver to collect RSSI measurements. We varied the distance between the sender and the receiver in different sites (ranging from 0.4 to 2 meters). Like WiAG [66], we collect gesture data from five positions. This is illustrated in Figure 7c where each data collection point gives the user's absolute position in centimeters (using the sender as the origin) as well as the orientation with respect to the receiver. The work of WiAG shows that a change in position of up to 31 cm or a change in orientation of up to 45 degrees has little impact on gesture recognition.

Participants. We recruited 100 volunteers (52 females) to participate in our experiments. As can be seen from Figure 9, our subjects have different heights, weights and somatotype, as indicated by various body mass index (BMI) values, and thus represent wide user groups.

Gait data. To collect CSI data for gait identification, each participant walked in one direction, from the entrance of



(a) Evaluation site layouts

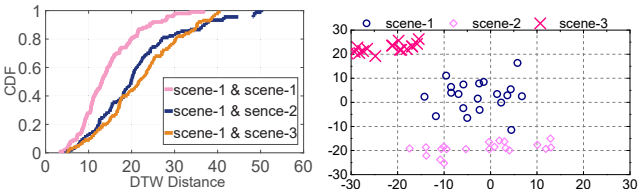
(b) Example wireless setup

(c) Positions for performing gestures

Figure 7: Experiment setup of evaluation site layouts (a), wireless setup (b) and gesture collection positions (c).

Table 4: Gestures used in our evaluation. These include primitive gestures and combinations of primitive gestures.

ID	Gesture	ID	Gesture	ID	Gesture	ID	Gesture	ID	Gesture	ID	Gesture	ID	Gesture
1	Flick	7	Pull	13	Hand moves down	19	Swipe right	25	Up-down	31	Cooking	37	Brushing teeth
2	Double flick	8	Throw	14	Hand moves up	20	Swipe left	26	Down-up	32	Studying	38	Put up the cigarette
3	Punch	9	Circle	15	Kick	21	Infinity	27	Sit down	33	Taking a bath	39	Put down the cigarette
4	Punch x 2	10	Dodge	16	Bowling	22	Left-right	28	Zoom In	34	Washing dishes	40	Inhale/exhale smoke
5	Level	11	Drag	17	Right	23	Right-left	29	Zoom Out	35	Turn on washing machines		
6	Push	12	Strike	18	Left	24	Head left	30	Eating	36	Play video games		



(a) (b)

Figure 8: DTW distances of gait measurements for *the same* person increase if samples are collected from different environments (a). The measurements of a user can be grouped into three clusters on a 2-d PCA space, each corresponds to where the data are gathered (b).

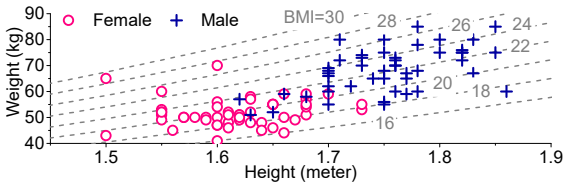


Figure 9: Heights, weights and BMIs of our participants who represent a wide range of user groups.

the room towards the other end of the room, to pass the two wireless devices at their natural speeds. We collected 20 wireless channel measurements per user per site, which yield in total 6,000 activity samples (100 users × 20 measurements per user × 3 sites).

Gesture data. We consider 40 gestures presented in prior work [2, 26, 32, 50, 60, 66, 77, 96]. The list of gestures is given in Table 4. These include primitive gestures like hand moves down, combinations of primitive gestures such as up-down (the hand first moves up and then moves down), and more complex patterns like putting up the cigarette [96]. Our evaluation targets a significantly larger number of gestures than any of the prior work seen to date, which typically was evaluated using a handful of gestures. We also stress that for gesture recognition, it is the number of target gestures matters as there is little difference for a gesture performed by different users. To collect gesture data, each user repeatedly perform each gesture 10 times in each site. This results in 1,200,000 activity samples (100 users × 40 gestures × 10 times per gesture × 2 channel metrics × 5 positions × 3 sites). Like other gesture recognition systems [2, 50, 66], we require the user to perform a preamble gesture – a punch in our case – to determine the location and orientation of the user.

Naming conventions. We use *collection_site - testing_site* to denote a cross-site scenario. For example, *s1-s2* means that we use the measurements collected in scene-1 to generate synthetic training examples for training and testing a sensing model for scene-2. Moreover, cross-site *transfer learning* is denoted as $(s_i - s_j, s_k)$, which is a pair of (i) an existing roaming model from sites i to j, and (ii) the target model for the new site k. For instance, $(s_1 - s_2, s_3)$ means that we first learn a roaming model, m_1, to translate WiFi training samples from scene-1 to scene-2, and then transfer the learned model, m_1, to build a new roaming model, m_2, to translate measurements from scene-1 to scene-3.

6.2 Evaluation Methodologies

Model Evaluation. We use *cross-validation* to evaluate our approach. The expert models are trained on the *synthetic* WiFi training measurements of the targets in a cross-site sensing scenario. We report the *geometric mean* accuracy across evaluation scenarios using the *top-1* score, i.e., we check if

the model's output of the highest probability matches the expected answer. There is minor change in results when using *top-2* and *top-3* scores. Performance variances are shown using min-max bars. Compared to the arithmetic mean, the geometric mean is widely considered as a better performance metric, as it can better minimize the impact of outliers [20].

Comparisons. We compare CrossSense against four state-of-the-art CSI-based sensing methods. These include WiWho [90] and WifiU [76] for gait identification, and WiAG [66] and WiG [26] for gesture recognition. We use the hand-tuned features of these competitive methods to construct sensing models. In addition to CSI, we also apply CrossSense to RSSI-based gesture recognition and compared it with TelepathicPhone [60]. We compare our roaming method to FitLoc [13], a WiFi signal translation approach for RSSI-based localization.

Implementation. CrossSense[2] is implemented in scikit-learn [49] and Matlab. It was trained on a high-performance server, but ran on a desktop PC with a dual-core 2.4GHz Corei5 CPU and 16GB of RAM running Ubuntu 16.04.

7 EXPERIMENTAL RESULTS

Highlights of our evaluation are as follows:

- CrossSense improves the classification accuracy of state-of-the arts from around 20% to over 80% under our settings (Section 7.1).
- CrossSense delivers consistently good performance regardless of the problem size (Section 7.2).
- Our roaming scheme allows existing WiFi sensing methods to work effectively across sites using a single set of WiFi training measurements (Section 7.3).
- Transfer learning reduces the number of samples needed for learning the roaming model by 4x (Section 7.5).
- We thoroughly evaluate CrossSense and provide detailed analysis on its working mechanisms.

7.1 Overall Performance

Figure 10 compares CrossSense against alternative approaches across training-testing-site pairs (i.e., using the WiFi training measurements collected from one site to build sensing models for another site). In all experiments, we make sure that the training and the testing datasets are different. As can be seen from the figure, existing approaches give disappointing results – less than 20% for most of the cases. This is because they cannot translate and re-use training measurements to build sensing models for a new environment. Further, the precision of existing approaches drops quickly as the number of target users or gestures increases. By contrast, CrossSense achieves the best accuracy for all evaluation scenarios, delivering an average accuracy of above 80% (and over 90% for CSI-based gesture recognition). CrossSense also gives less than 5% reduction in accuracy as the problem size increases from the smallest setting to the largest one. Compared to the over 2x drop in accuracy given by other schemes, CrossSense thus delivers the most reliable performance.

Figure 11 shows that CrossSense delivers consistent performance across site pairs. While CrossSense only uses one set of WiFi training measurements, its performance is comparable to or even better than other approaches when their sensing models are trained using samples collected from the target environment. As such, CrossSense allows scaling WiFi sensing to new environments and larger problem sizes with higher precision but at a lower cost.

Figure 12 summarizes how often an expert gives a correct classification across evaluation sites when targeting 100 users and 40 gestures. No single model correctly classifies more than 60% of the test cases, and the model capability varies from one environment to the other. The results emphasize that an universal model is unlikely to deliver good performance for large-scale and cross-site sensing problems.

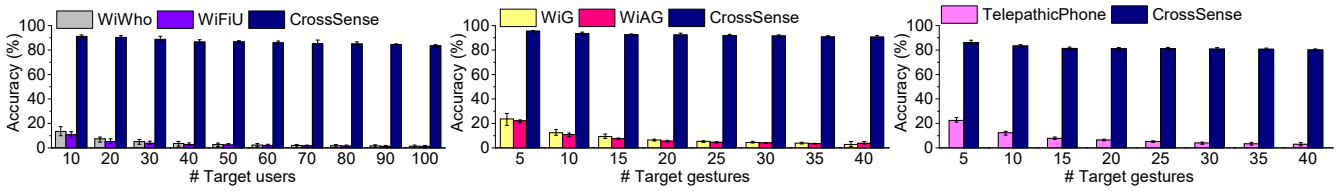
Finally, we note that a *theoretically perfect* expert selector for our collection of expert models would give an average accuracy of 94.5% and 98.5% respectively for CSI-based gait identification and gesture recognition. The results suggest that our chosen expert models are highly effective, which collectively lead to better sensing performance. If we can improve the accuracy of our expert selector, we can then further improve the accuracy of CrossSense.

7.2 Changing Problem Sizes

Figure 13 shows how the accuracy of CSI-based sensing changes as the problem size increases. To isolate the issue, we use the target's WiFi training measurements collected *from the deployment site* to build sensing models. We pick a given number of randomly chosen users and gestures as sensing targets, and repeat this process until all users and gestures are tested for at least once.

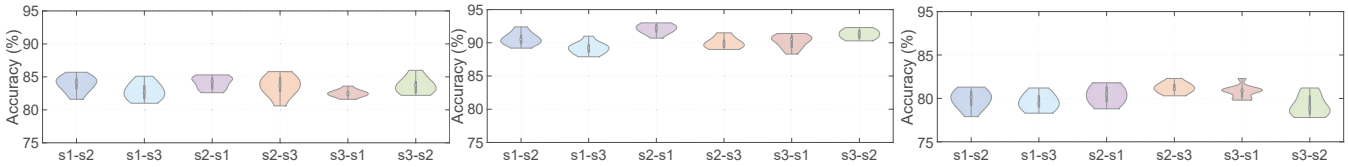
Using training measurements collected from the target environment does help existing approaches on a small problem size to deliver an accuracy of around 80%. However, performance of the competitive schemes degrades as the problem size increases. Their accuracy can drop by over 20% when targeting more than 50 users or 10 gestures. When moving from the smallest setting to the largest one, their accuracy can drop by over 50%. By contrast, CrossSense delivers not only the best accuracy for all testing scenarios but also consistent performance across test cases. The drop in accuracy for CrossSense is much smaller (less than 2% on average) when the problem size increases from the smallest to the largest one.

[2]Code and data are available at: https://goo.gl/4z4iXv.



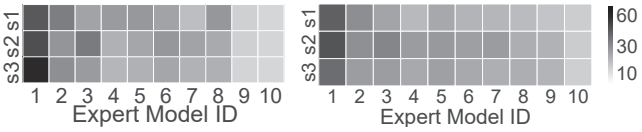
(a) CSI-based gait identification (b) CSI-based gesture recognition (c) RSSI-based gesture recognition

Figure 10: Overall performance per problem size across site pairs and cross-validation. CrossSense significantly outperforms all competitive approaches by delivering the best and the most robust performance.



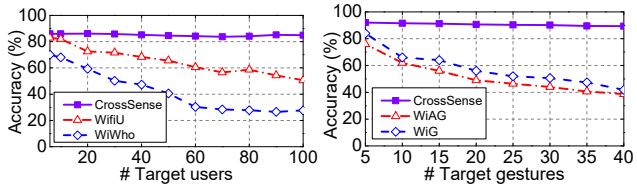
(a) CSI-based gait identification (b) CSI-based gesture recognition (c) RSSI-based gesture recognition

Figure 11: Accuracy distributions for CrossSense per training_data_collection-testing-site pair. The thick black line shows where 50% of the data lie. CrossSense delivers consistently good performance.



(a) CSI-based gait identification (b) CSI-based gesture recognition

Figure 12: How often (as percentages) an expert model in Table 3 correctly classifies an input across sites.



(a) CSI-based gait identification (b) CSI-based gesture recognition

Figure 13: CSI-based sensing performance with different numbers of target subjects. CrossSense provides consistently good performance.

7.3 Roaming WiFi Training Measurements

We now examine if the generated synthetic training samples are useful. To isolate the issue, we test all models on a *small-scale* problem of six target users and gestures at a time. Our roaming model is trained on data collected from users and gestures that are not used in testing. The roaming model is trained on 2,000 randomly chosen samples (1,000 measurements per site), and we repeat this process to ensure that each user and gesture is tested at least once.

Figure 14 compares the accuracy of each method with and without using our roaming scheme. Directly using training

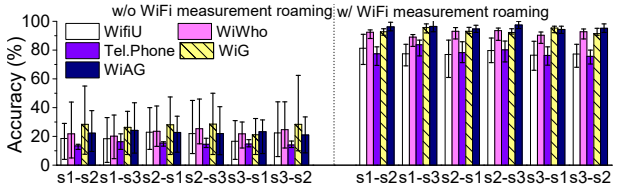


Figure 14: Cross-site sensing with and without using our roaming mechanism. Roaming training measurements helps existing methods to achieve comparable precision as if the training samples were collected from each deployment site.

samples of the target subjects collected from another site is a poor choice, giving only around 20% accuracy, which is only slightly better than making a random guess from the six possible options. Our roaming model gives a clear boost to cross-site WiFi sensing, improving the sensing accuracy by over 3.5x with an accuracy of above 75% (and over 80% for most of the cases). This level of performance is comparable to the accuracy when the model is built using training samples collected from the target environment. Here, we want to highlight that our roaming scheme is particularly useful when the sensing targets (e.g., visitors) constantly change, because the roaming model only needs to be trained once but can apply to unseen sensing targets.

7.4 CrossSense versus FitLoc

Wireless signal transfer has been exploited in prior work in indoor localization to translate wireless fingerprints between different areas within an environment. The transferring method employed by FitLoc [13] is the current state-of-the art. This experiment compares our roaming model

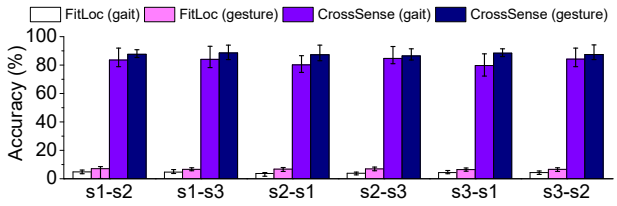


Figure 15: Comparison of the CrossSense's roaming model with the one used by FitLoc. The existing signal transferring method is ill-suited for cross-site WiFi sensing.

against FitLoc's transferring method. We use the translated (or synthetic) WiFi training examples collected from one site to build our expert models for another site. We then test the trained expert models on unseen WiFi measurements collected from the testing environment. Note that we use the same set of features to train the expert models, albeit that the translated samples given by FitLoc and CrossSense from the same raw WiFi training measurements are different. The experiment targets 100 users and 40 gestures.

Figure 15 shows that the method used by FitLoc is ill-suited for our problems, leading to a low classification accuracy – less than 10%. This is because the translated measurements produced by FitLoc have lost discriminative information and do not provide distinguishing features to separate sensing targets. By contrast, our roaming model can capture the essential characteristics of the wireless representations, and thus leads to better classification performance.

7.5 Transfer Learning

This experiment applies transferring learning (TL) to learn a roaming model for both a new site and a different sensing task. The accuracy is calculated by applying WiWho to six target users and WiAG to six gestures using cross-validation.

TL for a new site. Sub-figures a and b of Figure 16 show the results for using TL to learn a roaming model for an additional site. Figure 16a gives the mean accuracy across site permutations when the model is trained using 500 samples. Using TL, we achieve a comparable accuracy of over 80% and 90% respectively for gait identification and gesture recognition, but using only one quarter of the samples that would be required without TL. Figure 16b shows the average performance when the new roaming model is trained with different numbers of examples for site permutation $(s_1 - s_2, s_3)$. TL reduces the number of samples needed to reach a certain level of accuracy by approximately 4.5x, which in real terms means a saving of tens of hours for training data collection.

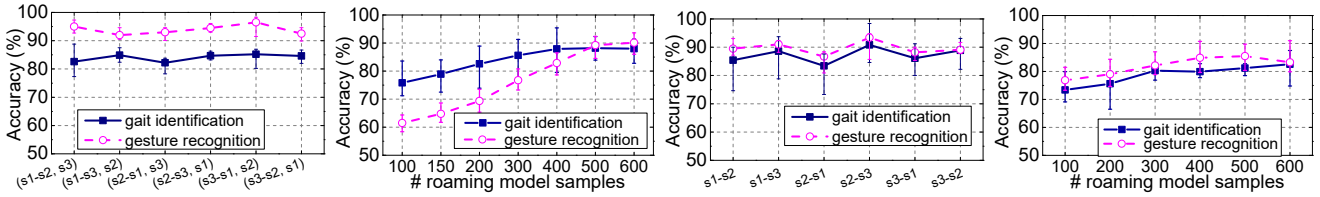
TL across tasks. Figures 16 c and d show the resulting accuracy when the roaming model is built from a model that was previously designed for another sensing task. Figure 16c gives the accuracy for different cross-site configurations. Figure 16d shows the accuracy achieved when different numbers of training samples are used to build a roaming model to transfer sensing model training samples from scene-1 to scene-2. The use of TL across problem domains can match the performance given by direct training, but requires nearly 4x fewer training examples.

Visualize network states. In an attempt to explain TL, Figure 17 depicts the internal state of the two roaming models – each translates a CSI measurement from scene-1 to scene-2 and scene-3, respectively. We use a heatmap to visualize the intensity of each neuron activation of a network layer. From the top to bottom, we begin with a specific input feature vector. As information flows through the network, the layers become progressively more specialized to the target deployment environment. The activations from the top to the bottom layers become increasingly diverge. The mean variance of activations across the two models increases by three folds from 0.70 at the first hidden layer to 2.25 at the output layer. TL allows us to reuse the early layers of the network to speed up learning when targeting a new site or task. We also see that the output feature vectors are significantly different from the input. This reinforces our claim that a WiFi training measurement collected from one site must be translated before it can be used in another site.

7.6 Adapt to Environmental Changes

Changes in the environment could have an impact on fine-grained sensing like gesture recognition. Prior work requires to recollect training data to update the sensing model if this happens. WiAG tackles the problem by using training samples collected from one location to create virtual samples for other target locations. Here we compare our roaming model against the virtual sample generation scheme of WiAG.

In the experiment, we added and removed a few stationary objects such as chairs and tables near the wireless devices. We repeated this process 10 times for each environment. This experiment was conducted over a period of three months. To adapt to the change, CrossSense first needs to collect 10 new samples from each target location from the changed environment (50 new samples for the five gesture locations in Figure 7c); it then uses the new samples to update the roaming model to generate *new* synthetic samples to refresh the sensing model. Collecting a gesture sample takes around 5 seconds, yielding a total of four minutes for 50 samples. Our *initial* roaming model is trained using 2,000 samples, which is a *one-off* cost. For WiAG, we use 500 *new* training samples to update its gesture translation model whenever the environment has changed. To isolate the impact of problem sizes, we use WiAG's classifier to recognize six target gestures. Our roaming model is trained on gestures that are not used in testing.



(a) Cross-site TL (b) Cross-site TL (s1 to s3) (c) Cross-task TL (d) Cross-task TL (s1 to s2)

Figure 16: Cross-site (a & b) and cross-task (c & d) transfer learning (TL). The accuracy is calculated by applying WiWho to six users and WiAG to six gestures. TL significantly reduces the number of samples required to learn a new roaming model.

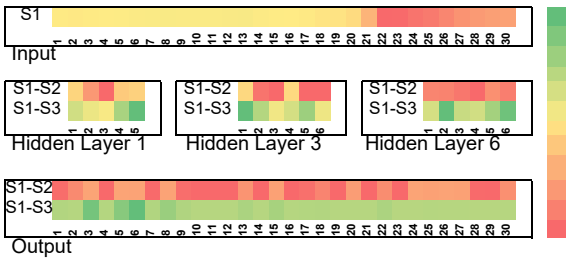
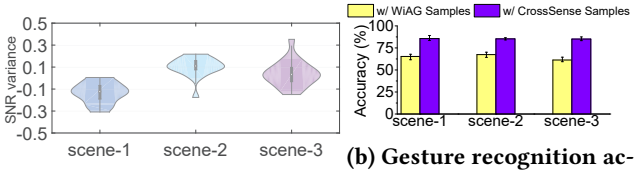


Figure 17: Internal neural states for two different roaming models when processing the same input. The activations in each layer increasingly diverge the lower down the network.



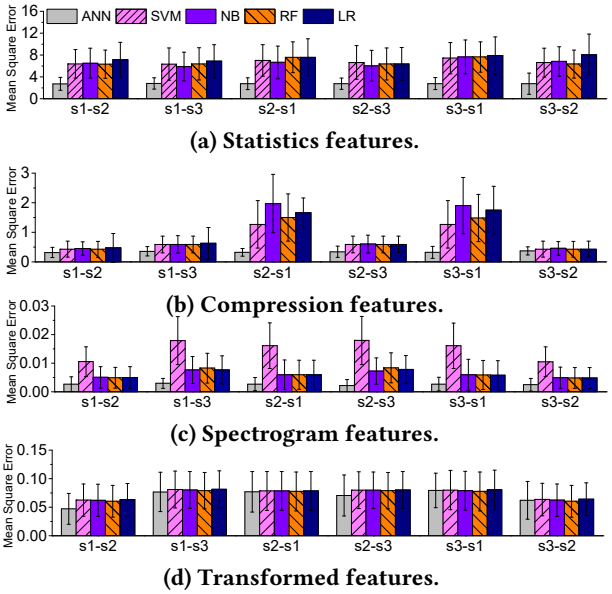
(a) SNR variances (b) Gesture recognition accuracy

Figure 18: SNR variances due to environmental changes (a) the resulting accuracy using synthetic samples produced by WiAG and CrossSense (b).

The violin diagram in Figure 18a shows the distribution of the signal-to-noise ratio (SNR) variance after the environment has changed. This is calculated from the participants' positions. Figure 18b shows that our roaming model uses 10x fewer new samples to refresh the sensing model, but leads to a better accuracy. This is because our ANN-based model together with TL can better capture the environmental impact to the wireless signal; and consequently, it generates synthetic training samples with a higher quality. To match the level of accuracy given by our roaming model, WiAG would require to collect at least 1,200 new samples every time in each of our testing environments. While CrossSense incurs slightly higher setup overhead, it can adapt to the change faster in a constantly evolving environment.

7.7 Roaming Model Analysis

We also compare our chosen roaming model against alternative modeling techniques and neural network structures.

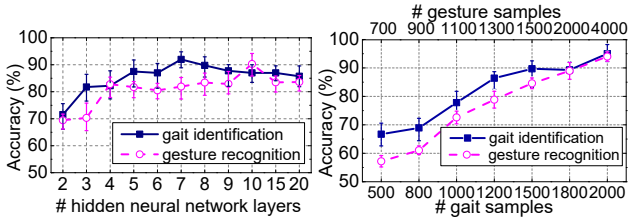


(a) Statistics features. (b) Compression features. (c) Spectrogram features. (d) Transformed features.

Figure 19: Comparisons of our ANN-based roaming model with other modeling techniques across feature sets for gait identification. Our model has the lowest mean square error.

Alternative modeling techniques. Figure 19 compares our ANN-based roaming model against four alternative regression model: support vector machines (SVM), naïve Bayes (NB), random forests (RF) and linear regression (LR), for each of the four feature sets listed in Table 1. We consider CSI-based gait identification and use the same process and data to train all models. The results are given in Figure 19. Since a measurement from one site can be mapped to multiple samples of the same subject collected from the other site, we calculate the mean square error for each possible mapping and show the range of errors on the min-max bars. The results suggest that the ANN model gives the lowest error across feature sets. Using an ANN also allows us to employ TL to reduce the cost for learning the roaming model.

Impact of neural layers and training samples. Figure 20a shows the classification accuracy when the roaming model is constructed with different numbers of hidden layers. The accuracy is calculated by applying WiWho and WiAG to six users and six gestures respectively using cross-validation;



(a) # hidden layers (b) # training samples.

Figure 20: Impact of the number of hidden neural layers (a) and training samples of the roaming model (b).

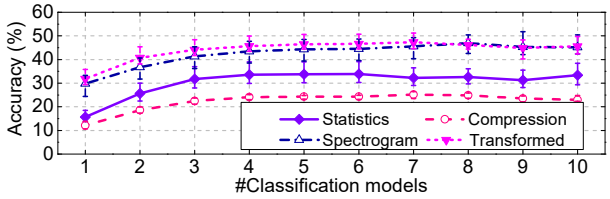


Figure 21: CSI-based gait identification when using different numbers of expert models to target 100 users.

and the roaming model is trained using 2,000 examples without TL. We observe that using 7 and 10 hidden layers give the best performance for gait identification and gesture recognition respectively. In this work, we choose to use a unified model structure with 7 hidden layers as it would require less training examples compared with the 10-hidden-layer alternative. Figure 20b reports a steady improvement in sensing model accuracy when using more examples to train the roaming model. Here we need more training data to build the gesture roaming model because gesture recognition needs to capture the subtle change of wireless signals at a finer-grained level. The figure also suggests that one can continuously improve the sensing model by providing more training samples over time (see also Section 7.9).

7.8 Impact of the Expert Model Size

Figure 21 shows how the number of expert models affects the gait identification accuracy on each feature set. In addition to the six models listed in Table 2, we also consider four additional techniques: ANNs, logical regression, stochastic gradient descent (SGD) and linear discriminant analysis (LDA). The accuracy is calculated on a per feature set basis by applying 10-fold cross-validation to the wireless measurements. We target 100 users, test all possible model mixtures, and collect the WiFi measurements from the target environment.

As we increase the number of expert models, there is an improvement in accuracy, confirming that a single model is insufficient. At the same time, we observe that the accuracy reaches a plateau when using six classifiers. We choose to use six classification techniques in the work because there is little gain in accuracy after that point to justify the increased cost in expert model training.

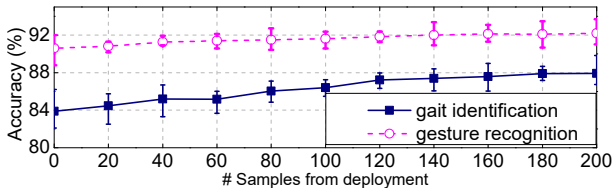


Figure 22: Using samples collected in the deployment environment to improve accuracy.

7.9 Continuous Learning

Figure 22 shows that the sensing performance can be continuously improved using samples collected from the deployment environment. The results are obtained by applying our mixture-of-experts method to 100 users and 40 gestures. The accuracy of the initial sensing model (learned using synthetic samples) can be improved by 4% for gait identification and 2% for gesture identification as we adding more samples obtained from the deployment site to the training dataset.

8 DISCUSSIONS AND LIMITATIONS

Naturally there is room for further work and improvements. We discuss a few points here.

Training cost. While our approach significantly reduces the cost of training data collection, it does not eliminate it. To further reduce the cost of learning the roaming model, one can employ a continual learning strategy, i.e., starting with a modest accurate model and then using the data collected in target environment to continuously improve the model over time (see Section 7.9). Active learning [15, 43, 44] can also be employed to direct the attention to collect samples that are most likely to improve the accuracy of the model.

Regression-oriented problems. CrossSense is evaluated on classification-based sensing problems. For regression-based problems, e.g., tracking and localization [84], we would need to replace our current classifier-based expert models with regression models. This may also require employing new features, but our methods for training measurement roaming, feature engineering and expert model tuning remain applicable.

Deep learning based feature representation. Recently, deep neural networks [17, 86] are shown to be powerful in extracting feature representations. However, they require many more training examples than that we use, which would incur prohibitive overhead because of user involvement. On the other hand, deep learning can be used within a continual- and transfer-learning framework to replace some of our expert models or obtain signal embeddings [21] as features.

Other sensing platforms. We believe CrossSense can be applied to radio frequency (RF) based sensing [6, 70, 83] and

the extension is our future work. It would also be interesting to see if CrossSense can be combined with specialized sensing hardware [5, 7, 37] to provide higher precision.

Limitations. CrossSense is evaluated in a controlled environment (see Section 6.1). Limitations of our evaluation include using a single-link wireless setup and a single testbed across environments, a relatively short distance (within two meters) between the wireless transmitter and the receiver, only a single person is presented (i.e., no interference from other people around), and the changes in one environment over time are minor (e.g., furniture movements) rather than drastic (e.g., opening or closing doors). In future work, we plan to further assess the performance of CrossSense in more complex settings.

9 RELATED WORK

Recent years have witnessed a growing interest in exploiting wireless signals for tasks like gait identification [27, 76, 90, 92], gesture recognition [8, 26, 32, 40, 42, 53–57, 62, 66, 71, 93], localization [11, 63, 64, 69, 84, 88], health and risk assessment [4, 7, 25, 36, 38, 45, 47, 48, 68], activity detection [10, 14, 58, 59, 74, 75, 77, 80, 81, 89], human detection [18, 82, 97, 98] and emotion recognition [94]. Indeed, wireless sensing has moved from an a research niche [78] to a mainstream activity.

Wireless sensing maps a wireless measurement to an output. The dominant approach in most sensing applications employs a classifier that requires collecting the targets' training data from each deployment environment. WiAG [66] tackles the problem through generating virtual training samples for different locations in a single environment. However, WiAG is designed for gesture recognition but not cross-site sensing. Our work builds upon WiAG, utilizing its direction and location identification method to offer a generic framework for cross-site sensing. Other approaches use a so-called parametric approach whereby certain characteristics of the signal are linked with target activities. Parametric techniques are mostly limited to specific application areas, with gesture recognition being a prominent example [2, 32, 50], and a small set of target activities. It is worth mentioning that an alternative approach for cross-site sensing is to leverage the frequency modulated carrier wave (FMCW) to track activities [3, 6, 41]. Such an approach can generalize to different environments and handle multipath. However, it is not compatible with the commercial-off-the-self wireless devices.

As we have shown in the paper, another significant drawback of prior sensing methods is that they typically only work on a small problem size. We address this problem by combing and selecting multiple sensing models.

Transfer learning is shown to be useful in activities recognition [16], localization [46], crowdsourced mobile activity learning [95], and human activity recognition using sensor data [72]. Previous studies use transfer learning to translate training data, features or fine-tuning models for mobile sensor data. CrossSense builds on these past foundations to enable cross-site and cross-domain WiFi sensing.

Our mixture-of-experts approach is a form of ensemble learning [91]. Ensemble learning has been used in *video-based* gesture recognition [79] and gait identification [22], face recognition [23], information filtering [61], and other optimization tasks [19, 39]. However, no work so far has employed ensemble learning for WiFi sensing and this work is the first to do so. We would like to stress that the goal of CrossSense is not to advance ensemble learning; instead, it uses the technique together with signal transferring methods to design a general learning framework for wireless sensing.

10 CONCLUSIONS

This paper has presented CrossSense, a framework to enable WiFi sensing to work effectively across deployment sites and to target a large number of sensing targets. To reduce the cost and human involvement for cross-site sensing, CrossSense employs a machine learning model to generate, from a single set of WiFi training measurements, synthetic training samples for each deployment environment. To enable WiFi sensing to scale up to a larger problem size, CrossSense integrates a mixture-of-experts based sensing approach. It determines at runtime, the best sensing model out of a collection of models (experts), as there is no "one-size-fits-all" universal best model. Such an approach provides a mechanism to gracefully add additional new sensing models to target a wider range of sensing scenarios and tasks. We demonstrate the effectiveness of CrossSense by applying it to gait identification and gesture recognition and thoroughly evaluate it using over 1.2 million WiFi activity samples. Compared to prior work, CrossSense delivers the best and the most reliable performance across evaluation scenarios, and can work efficiently on problem sizes that are significantly bigger than that the current approaches can effectively handle.

ACKNOWLEDGEMENT

We are grateful to our shepherd and the anonymous reviewers for their insightful feedback. This work was partially supported by the National Natural Science Foundation of China under grant agreements 61672427, 61672428 and 61772422; the China Scholarship Council (201806970007); the Science and Technology Innovation Team Support Program of Shaanxi Province, China (2018TD-O26); the UK EPSRC under grant agreements EP/M01567X/1 (SANDeRs) and EP/M015793/1 (DIVIDEND); and the Royal Society International Collaboration Grant (IE161012). We thank to JD Cloud for the use of their computing servers.

REFERENCES

[1] Heba Abdelnasser et al. 2015. UbiBreathe: A Ubiquitous non-Invasive WiFi-based Breathing Estimator. In *MobiHoc*.

[2] Heba Abdelnasser et al. 2015. WiGest: A ubiquitous WiFi-based gesture recognition system. In *INFOCOM*.

[3] Fadel Adib et al. 2014. 3D tracking via body radio reflections. In *NSDI*.

[4] Fadel Adib et al. 2014. Demo: real-time breath monitoring using wireless signals. *MobiCom*.

[5] Fadel Adib et al. 2015. Capturing the Human Figure Through a Wall. *ACM Transactions on Graphics* (2015).

[6] Fadel Adib et al. 2015. Multi-Person Localization via RF Body Reflections. In *NSDI*.

[7] Fadel Adib et al. 2015. Smart Homes That Monitor Breathing and Heart Rate. In *CHI*.

[8] Fadel Adib and Dina Katabi. 2013. See through walls with WiFi!. In *SIGCOMM*.

[9] Kamran Ali et al. 2015. Keystroke Recognition Using WiFi Signals. In *MobiCom*.

[10] Kamran Ali et al. 2017. Recognizing keystrokes using WiFi devices. *IEEE Journal on Selected Areas in Communications* (2017).

[11] Heba Aly and Moustafa Youssef. 2015. An Analysis of Device-Free and Device-Based WiFi-Localization Systems. In *IJACI*.

[12] Anthony Bagnall et al. 2015. Time-Series Classification with COTE: The Collective of Transformation-Based Ensembles. *IEEE Transactions on Knowledge and Data Engineering* (2015).

[13] L. Chang et al. 2017. FitLoc: Fine-Grained and Low-Cost Device-Free Localization for Multiple Targets Over Various Areas. *IEEE/ACM Transactions on Networking* (2017).

[14] Bo Chen et al. 2015. Tracking Keystrokes Using Wireless Signals. In *MobiSys*.

[15] David A Cohn et al. 1996. Active learning with statistical models. *Journal of Artificial Intelligence Research* (1996).

[16] Diane Cook et al. 2013. Transfer learning for activity recognition: A survey. *Knowledge and Information Systems* (2013).

[17] Chris Cummins et al. 2017. End-to-end deep learning of optimization heuristics. In *PACT*.

[18] Simone Di Domenico et al. 2016. A Trained-once Crowd Counting Method Using Differential WiFi Channel State Information. In *International on Workshop on Physical Analytics*.

[19] Murali Krishna Emani and Michael O'Boyle. 2015. Celebrating Diversity: A Mixture of Experts Approach for Runtime Mapping in Dynamic Environments. In *PLDI*.

[20] Wolfgang Ertel. 1994. On the definition of speedup. In *International Conference on Parallel Architectures and Languages Europe*.

[21] Daniel Garcia-Romero et al. 2017. Speaker diarization using deep neural network embeddings. In *ICASSP*.

[22] Y. Guan et al. 2015. On Reducing the Effect of Covariate Factors in Gait Recognition: A Classifier Ensemble Method. *IEEE Transactions on Pattern Analysis and Machine Intelligence* (2015).

[23] Srinivas Gutta and Harry Wechsler. 1996. Face recognition using hybrid classifier systems. In *ICNN*.

[24] Dan Halperin. [n. d.]. Linux-80211n-csitool. http://dhalperi.github.io/linux-80211n-csitool/.

[25] Chunmei Han et al. 2014. WiFall: Device-free fall detection by wireless networks. In *INFOCOM*.

[26] Wenfeng He et al. 2015. WiG: WiFi-Based Gesture Recognition System. In *ICCCN*.

[27] Feng Hong et al. 2016. WFID: Passive Device-free Human Identification Using WiFi Signal. In *MOBIQUITOUS*.

[28] Chen-Yu Hsu et al. 2017. Extracting Gait Velocity and Stride Length from Surrounding Radio Signals. In *CHI*.

[29] Zhiping Jiang et al. 2013. Rejecting the attack: Source authentication for wi-fi management frames using csi information. In *INFOCOM*.

[30] Michael I. Jordan and Robert A. Jacobs. 1994. Hierarchical Mixtures of Experts and the EM Algorithm. *Neural Comput.* (1994).

[31] Ehud D Karnin. 1990. A simple procedure for pruning back-propagation trained neural networks. *IEEE Transactions on Neural Networks* (1990).

[32] Bryce Kellogg et al. 2014. Bringing gesture recognition to all devices. In *NSDI*.

[33] Jin Kun et al. 2017. Vivisnoop: Someone is snooping your typing without seeing it!. In *CNS*.

[34] Mengyuan Li et al. 2016. When CSI Meets Public WiFi: Inferring Your Mobile Phone Password via WiFi Signals. In *CCS*.

[35] Hongbo Liu et al. 2014. Practical User Authentication Leveraging Channel State Information (CSI). In *ASIA CCS*.

[36] Xuefeng Liu et al. 2015. Wi-Sleep: Contactless Sleep Monitoring via WiFi Signals. In *RTSS*.

[37] Bastien Lyonnet et al. 2010. Human gait classification using microdoppler time-frequency signal representations. In *RadarConf*.

[38] B. Mager et al. 2013. Fall detection using RF sensor networks. In *PIMRC*.

[39] Vicent Sanz Marco et al. 2017. Improving spark application throughput via memory aware task co-location: a mixture of experts approach. In *Middleware*.

[40] Pedro Melgarejo et al. 2014. Leveraging directional antenna capabilities for fine-grained gesture recognition. In *UbiComp*.

[41] Pavlo Molchanov et al. 2015. Short-range FMCW monopulse radar for hand-gesture sensing. In *RadarConf*.

[42] Rajalakshmi Nandakumar et al. 2014. Wi-Fi Gesture Recognition on Existing Devices. *Arxiv* (2014).

[43] William Ogilvie et al. 2014. Fast automatic heuristic construction using active learning. In *LCPC*.

[44] William F Ogilvie et al. 2017. Minimizing the cost of iterative compilation with active learning. In *CGO*.

[45] Sameera Palipana et al. 2018. FallDeFi: Ubiquitous Fall Detection using Commodity Wi-Fi Devices. In *UbiComp*.

[46] Sinno Jialin Pan et al. 2008. Transfer learning via dimensionality reduction.. In *AAAI*.

[47] Neal Patwari et al. 2011. Monitoring Breathing via Signal Strength in Wireless Networks. In *IEEE Transactions on Mobile Computing*.

[48] Neal Patwari et al. 2013. Breathfinding: A Wireless Network That Monitors and Locates Breathing in a Home. *IEEE Journal of Selected Topics in Signal Processing* (2013).

[49] Fabian Pedregosa et al. 2011. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research* (2011).

[50] Qifan Pu et al. 2013. Whole-home gesture recognition using wireless signals. In *MobiCom*.

[51] Kun Qian et al. 2017. Inferring Motion Direction using Commodity Wi-Fi for Interactive Exergames. In *CHI*.

[52] Stan Salvador and Philip Chan. 2007. Toward accurate dynamic time warping in linear time and space. *Intelligent Data Analysis* (2007).

[53] Muhammad Shahzad et al. 2017. Behavior Based Human Authentication on Touch Screen Devices Using Gestures and Signatures. *IEEE Transactions on Mobile Computing* (2017).

[54] Jiacheng Shang and Jie Wu. 2017. A Robust Sign Language Recognition System with Multiple Wi-Fi Devices. In *MobiArch*.

[55] Jiacheng Shang and Jie Wu. 2017. A Robust Sign Language Recognition System with Sparsely Labeled Instances Using Wi-Fi Signals. In *MobiHoc*.

[56] Longfei Shangguan et al. 2017. Enabling Gesture-based Interactions with Objects. In *MobiSys*.

[57] Cong Shi et al. 2017. Smart User Authentication through Actuation of Daily Activities Leveraging WiFi-enabled IoT. In *MobiHoc*.

[58] Stephan Sigg et al. 2013. Leveraging RF-channel fluctuation for activity recognition: Active and passive systems, continuous and RSSI-based signal features. In *MoMM*.

[59] Stephan Sigg et al. 2014. RF-Sensing of Activities from Non-Cooperative Subjects in Device-Free Recognition Systems Using Ambient and Local Signals. *IEEE Transactions on Mobile Computing* (2014).

[60] Stephan others Sigg. 2014. The telepathic phone: Frictionless activity recognition from WiFi-RSSI. In *PerCom*.

[61] Hichem Snoussi and Cédric Richard. 2006. Ensemble learning online filtering in wireless sensor networks. In *ICCS*.

[62] Sheng Tan and Jie Yang. 2016. WiFinger: leveraging commodity WiFi for fine-grained finger gesture recognition. In *MobiHoc*.

[63] Xiaohua Tian et al. 2017. Performance Analysis of RSS Fingerprinting Based Indoor Localization. *IEEE Transactions on Mobile Computing* (2017).

[64] Xiaohua Tian et al. 2018. Improve Accuracy of Fingerprinting Localization with Temporal Correlation of the RSS. *IEEE Transactions on Mobile Computing* (2018).

[65] H. T. T. Truong et al. 2014. Comparing and fusing different sensor modalities for relay attack resistance in Zero-Interaction Authentication. In *PerCom*.

[66] Aditya Virmani and Muhammad Shahzad. 2017. Position and Orientation Agnostic Gesture Recognition Using WiFi. In *MobiSys*.

[67] Guanhua Wang et al. 2014. We Can Hear You with Wi-Fi!. In *MobiCom*.

[68] Hao Wang et al. 2016. Human respiration detection with commodity wifi devices: do user location and body orientation matter?. In *UbiComp*.

[69] Ju Wang et al. 2016. LiFS: low human-effort, device-free localization with fine-grained subcarrier information. In *MobiCom*.

[70] Ju Wang et al. 2017. TagScan: Simultaneous Target Imaging and Material Identification with Commodity RFID Devices. In *MobiCom*.

[71] J. Wang et al. 2018. Device-free Wireless Sensing in Complex Scenarios Using Spatial Structural Information. *IEEE Transactions on Wireless Communications* (2018).

[72] Jindong Wang et al. 2018. Stratified Transfer Learning for Cross-domain Activity Recognition. In *PerCom*.

[73] Jue Wang and Dina Katabi. 2013. Dude, where's my card?: RFID positioning that works with multipath and non-line of sight. In *SIGCOMM*.

[74] Liang Wang et al. 2016. Toward a Wearable RFID System for Real-Time Activity Recognition Using Radio Patterns. *IEEE Transactions on Mobile Computing* (2016).

[75] Wei Wang et al. 2015. Understanding and Modeling of WiFi Signal Based Human Activity Recognition. In *MobiCom*.

[76] Wei Wang et al. 2016. Gait recognition using wifi signals. In *UbiComp*.

[77] Yan Wang et al. 2014. E-eyes: device-free location-oriented activity identification using fine-grained wifi signatures. In *MobiCom*.

[78] Kristen Woyach et al. 2006. Sensorless sensing in wireless networks: Implementation and measurements. In *WiOpt*.

[79] Di Wu et al. 2012. One shot learning gesture recognition from rgbd images. In *CVPRW*.

[80] Dan Wu et al. 2016. WiDir: walking direction estimation using wireless signals. In *UbiComp*.

[81] Dan Wu et al. 2017. Device-Free WiFi Human Sensing: From Pattern-Based to Model-Based Approaches. *IEEE Communications Magazine* (2017).

[82] Wei Xi et al. 2014. Electronic frog eye: Counting crowd using wifi. In *INFOCOM*.

[83] Fu Xiao et al. 2018. One More Tag Enables Fine-Grained RFID Localization and Tracking. *IEEE/ACM Transactions on Networking* (2018).

[84] Jiang Xiao et al. 2013. Pilot: Passive device-free indoor localization using channel state information. In *ICDCS*.

[85] Tong Xin et al. 2016. FreeSense: Indoor Human Identification with Wi-Fi Signals. In *GlobeCom*.

[86] Shuochao Yao et al. 2017. DeepSense: A Unified Deep Learning Framework for Time-Series Mobile Sensing Data Processing. In *WWW*.

[87] Jason Yosinski et al. 2014. How Transferable Are Features in Deep Neural Networks?. In *NIPS*.

[88] Moustafa Youssef. 2016. CoSDEO 2016 Keynote: A decade later? Challenges: Device-free passive localization for wireless environments. In *PerCom Workshops*.

[89] Yunze Zeng et al. 2014. Your AP knows how you move:fine-grained device motion recognition through WiFi. In *HotWireless*.

[90] Yunze Zeng et al. 2016. WiWho: WiFi-Based Person Identification in Smart Spaces. In *IPSN*.

[91] Cha Zhang and Yunqian Ma. 2012. *Ensemble machine learning: methods and applications*. Springer.

[92] Jin Zhang et al. 2016. WiFi-ID: Human Identification Using WiFi Signal. In *DCOSS*.

[93] Ouyang Zhang and Kannan Srinivasan. 2016. Mudra: User-friendly Fine-grained Gesture Recognition using WiFi Signals. In *CoNEXT*.

[94] Mingmin Zhao et al. 2017. Emotion recognition using wireless signals. In *MobiCom*.

[95] Zhongtang Zhao et al. 2011. Cross-people mobile-phone based activity recognition. In *IJCAI*.

[96] Xiaolong Zheng et al. 2016. Smokey: Ubiquitous smoking detection with commercial WiFi infrastructures. In *INFOCOM*.

[97] Zimu Zhou et al. 2013. Towards omnidirectional passive human detection. In *INFOCOM*.

[98] Hai Zhu et al. 2017. R-TTWD: Robust Device-free Through-The-Wall Detection of Moving Human with WiFi. *IEEE Journal on Selected Areas in Communications* (2017).