Augmenting User Identification with WiFi Based Gesture Recognition

MUHAMMAD SHAHZAD, North Carolina State University, USA
SHAOHU ZHANG, North Carolina State University, USA

Over the last few years, researchers have proposed several WiFi based gesture recognition systems that can recognize predefined gestures performed by users at runtime. As most environments are inhabited by multiple users, the true potential of WiFi based gesture recognition can be unleashed only when each user can independently define the actions that the system should take when the user performs a certain predefined gesture. To enable this, a gesture recognition system should not only be able to recognize any given predefined gesture, but should also be able to identify the user that performed it. Unfortunately, none of the prior WiFi based gesture recognition systems can identify the user performing the gesture. In this paper, we propose WiID, a WiFi and gesture based user identification system that can identify the user as soon as he/she performs a predefined gesture at runtime. WiID integrates with the WiFi based gesture recognition systems as an add-on module whose sole objective is to identify the users that perform the predefined gestures. The design of WiID is based on our novel result which states that the time-series of the frequencies that appear in WiFi channel's frequency response while performing a given gesture are different in the samples of that gesture performed by different users, and are similar in the samples of that gesture performed by the same user. We implemented and extensively evaluated WiID in a variety of environments using a comprehensive data set comprising over 25,000 gesture samples.

CCS Concepts: • **Human-centered computing → Gestural input**;

Additional Key Words and Phrases: User identification, Gesture recognition, WiFi

ACM Reference Format:
Muhammad Shahzad and Shaohu Zhang. 2018. Augmenting User Identification with WiFi Based Gesture Recognition. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.* 2, 3, Article 134 (September 2018), 27 pages. https://doi.org/10.1145/3264944

1 INTRODUCTION

1.1 Background and Motivation

Over the past few years, WiFi based sensing has gained significant popularity and researchers have proposed several WiFi based systems that recognize human gestures [10, 13, 14, 24, 26, 30–33, 36]. The motivation behind the majority of the work on WiFi based gesture recognition is to improve the quality and experience of living of users in smart environments by enabling them to naturally interact, communicate, and control the smart devices and the ubiquitous computing that are increasingly getting embedded into our environments. The fundamental principle that enables human gesture recognition using WiFi is that the WiFi channel metrics, such as channel state information (CSI) and received signal strength (RSS), change when a user moves in a wireless environment. The patterns of change in these WiFi channel metrics are unique for different gestures. WiFi based gesture recognition systems first learn these patterns of change using machine learning techniques for each predefined gesture and then recognize them when a user performs them at runtime.

Authors' addresses: Muhammad Shahzad, North Carolina State University, 890 Oval Drive, Raleigh, NC, USA, 27606, mshahza@ncsu.edu; Shaohu Zhang, North Carolina State University, 890 Oval Drive, Raleigh, NC, USA, 27606, szhang42@ncsu.edu.

2474-9567/2018/09-ART134 $15.00
https://doi.org/10.1145/3264944

While most prior WiFi based gesture recognition systems can recognize any predefined gesture at runtime irrespective of the user that performs it, unfortunately, the prior WiFi based gesture recognition systems cannot identify *which* user performed the gesture. As most environments are usually inhabited by multiple users, the true potential of WiFi based gesture recognition can be unleashed only when each user can independently define the actions that the system should take when he/she performs a certain predefined gesture. For example, assuming that the various devices in an environment are connected to and can be controlled through a centralized control center (such as the Microsoft Home OS [17], Contiki [4], House Operating System [6], or Ambient OS [2]) and that a WiFi based gesture recognition and user identification system is implemented on that centralized control center, one user may be interested in controlling the ambience related devices in the environment while another user may be interested in controlling the multimedia devices in that same environment using the same set of simple gestures. This can be possible only when, along with recognizing any given gesture, the centralized control center can also identify the user that performed it. Another area where augmenting user identification with WiFi based gesture recognition finds application is soft access control. For example, the gestures from a child in a home should not be allowed to turn on the TV at the times of the day when the child is not allowed to watch TV, while the gestures from a parent should be allowed to turn on the TV at any time of the day.

1.2 Problem Statement

In this paper, our objective is to take WiFi based gesture recognition to its next step of evolution where in addition to recognizing any given predefined gesture performed by a user, the WiFi based system should further be able to identify the user that performed that gesture. The system should be able to identify the user directly from the gesture and should not require the user to perform any additional movements. Moreover, the system should be implementable on commodity WiFi devices without requiring any specialized hardware.

1.3 Proposed Approach

A seemingly obvious approach to enable different users to control different aspects of their environment is to simply ask them to use different gestures, which will eliminate the need for identifying the users altogether. This approach, however, is neither scalable nor practical. The scalability issue in this approach arises because if each user was to use a different set of gestures, then the complexity, duration, and the number of training samples of each gesture must increase significantly to enable the machine learning based methods to distinguish between such a large set of gestures. Thus, while this naive approach is suitable when the number of unique gestures that each user should be able to perform is very less (such as one or two), it is not well-suited if the number of gestures that each user should be able to perform increases (such as 3 or more gestures per user for 5 or more users). The impracticality issue in this approach arises because all users must coordinate with each other to ensure that no two users are using similar gestures; and if they are, they are not asking the system to take different actions in response to the same gesture. In many environments, such as in offices and even in homes, it is non-trivial to not only achieve such coordination due to busy schedules but also to arrive at a decision in the case of conflicts about who should use a certain gesture if multiple users desire to use it. This, in turn, can lead to social friction and raising social barriers, and thus negatively impact the occupants' quality and experience of living; an outcome that is quite the opposite of the motivation behind the WiFi based gesture recognition systems.

In this paper, we propose WiID, a WiFi and gesture based user IDentification system that can identify the user as soon as he/she performs a predefined gesture at runtime. We emphasize here that the objective of WiID is not to recognize the predefined gestures; its objective is to identify the user who performs any given predefined gesture. WiID assumes that a WiFi based gesture recognition system, such as WiAG [33] or CARM [35], is already in place to recognize the gestures. As soon as the gesture recognition system detects and recognizes a gesture, WiID springs into action and processes the WiFi channel metrics of this recognized gesture to identify the user that performed this gesture. WiID integrates with WiFi based gesture recognition systems as an add-on module whose sole objective is to identify the users that perform the predefined gestures.

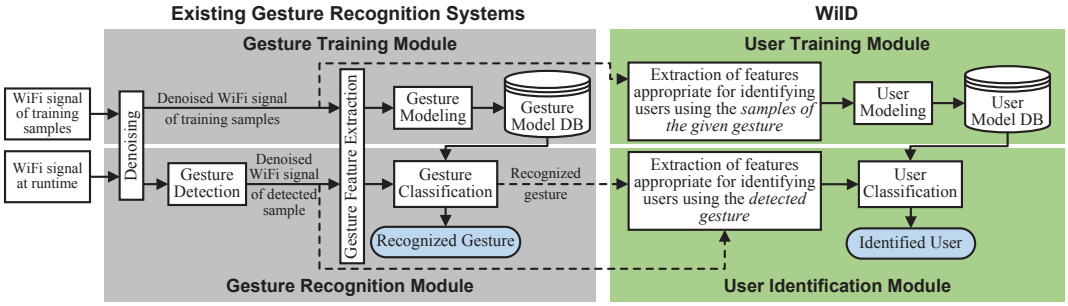


Fig. 1. Block diagram of WiID and how it augments with a WiFi based gesture recognition system

Figure 1 shows the block diagram demonstrating the general work flow of typical WiFi based gesture recognition systems and how WiID integrates with them as an add-on module. Next, we first briefly describe the general work flow of typical WiFi based gesture recognition systems and then provide an overview of how WiID integrates with them. After that, we describe the intuition that WiID leverages to identify the user as soon as the gesture recognition system detects and recognizes a gesture.

Overview of Typical Gesture Recognition Systems. To recognize any predefined gesture, a WiFi based gesture recognition system first needs a classification model of that gesture. For this, it collects multiple training samples of that gesture from users, removes noise from the WiFi channel metrics of these samples, and extracts those features from the denoised channel metrics that are appropriate for distinguishing between different predefined gestures. Next, it uses these features from all training samples of all predefined gestures and uses appropriate machine learning methods to build classification models of those gestures. This process of generating the classification models is shown in the block diagram inside the top gray box in Figure 1. To recognize the predefined gestures at runtime, the gesture recognition system continuously monitors the WiFi channel metrics, removes noise from them, and waits until an occupant performs a gesture. As soon as it detects that an occupant has performed a gesture, it takes the denoised WiFi channel metrics captured during the start and end times of the detected gesture, extracts the same features from them that it extracted from the training samples, evaluates these features against all classification models, and recognizes the detected gesture as that predefined gesture whose classification model provides the highest match. This process of detecting and recognizing the gestures at runtime is shown in the block diagram inside the bottom gray box in Figure 1.

Overview of our User Identification System. To identify any given user from the predefined gestures, WiID needs a classification model of that user for each predefined gesture. To generate these classification models, WiID takes the denoised WiFi channel metrics of each training sample of each predefined gesture from each user, extracts features that are appropriate for distinguishing between different users, and generates a classification model of each user for each gesture. This process is summarized in the block diagram inside the top green box in Figure 1. Note that WiID does not require users to provide any new training samples. It simply reuses the training samples that the WiFi based gesture recognition system already collected. At runtime, as soon as the gesture recognition system detects and recognizes a gesture, WiID takes the denoised WiFi channel metrics between the start and end times of the detected gesture as input along with the decision made by the gesture recognition system. Based on this decision, it extracts the same features from the denoised WiFi channel metrics that it extracted from the training samples of the recognized gesture from all users, evaluates these features against the classification models of all users for that gesture, and identifies the user as the one whose classification model provides the highest match. This process is shown inside the bottom green box in Figure 1. Note that in this entire process of generating classification models for different users and identifying them at runtime when the users perform gestures, WiID did not propose any modifications to the WiFi based gesture recognition systems.

Intuition Behind WiID. The method that WiID employs to identify users is based on the combination of a *hypothesis* about user behaviors and a *theoretical result* about the impact of user movements on channel state information (CSI), a well known WiFi channel metric. The hypothesis is that different users have different behaviors of performing any given gesture but the behavior of any given user of performing the given gesture stays consistent over time. In other words, the time-series of the speeds with which the users move their limbs while performing any given gesture are different in the samples of that gesture performed by different users, and are similar in the samples of that gesture performed by the same user. We will study the validity of this hypothesis through a comprehensive measurement study. The theoretical result states that the frequencies that appear in the CSI measurements are directly proportional to the speeds with which a human limb moves while performing a gesture [35]. By combining the hypothesis with this theoretical result, we obtain the following result: the time-series of the frequencies that appear in the CSI measurements while performing a given gesture are different in the samples of that gesture performed by different users, and are similar in the samples of that gesture performed by the same user. Therefore, WiID first extracts the time-series of the frequencies from the CSI measurements of any given sample using signal processing techniques, and then converts this time-series of frequencies into the time-series of speeds. Next, it calculates the values of appropriate features from this speed time-series such that these features have distinguishing values across the samples from different users but consistent values across the samples from the same user. Finally, WiID uses these features to identify the user.

1.4 Technical Challenges

In designing WiID, we faced several technical challenges. For example, the first technical challenge is to extract the time-series of frequencies that are introduced in the CSI measurements by the moving limb that performs any given gesture sample. For this, WiID applies short time fourier transform (STFT) on the time-series of the denoised CSI measurements and obtains a spectrogram. A spectrogram of a time-series is essentially the transformation of that time-series from time domain to time-frequency domain [7]. As per the theoretical result mentioned earlier, the frequencies that are introduced in the CSI measurements due to the movement of the limb show a higher magnitude compared to the remaining frequencies. WiID extracts these higher magnitude frequencies from this spectrogram using contour-extraction technique and thus, obtains the desired time-series of frequencies introduced by the moving limb.

The second challenge is to segment the speed time-series (derived from the time-series of frequencies) in samples of any given gesture into multiple smaller time-series such that the user has consistent and distinguishing behavior within those smaller time-series. We refer to these smaller time-series as subseries. It is challenging to determine the number of subseries that a speed time-series should be segmented into, the starting point of each subseries, and the duration of each subseries. On one hand, if the time duration of a subseries is too short, then the user may not have consistent behavior for that subseries when performing a given gesture multiple times. On the other hand, if the time duration of a subseries is too large, then the distinguishing information from the features extracted from this subseries may get averaged out. The time duration of different subseries should not be all equal either because at different locations of a gesture, users have consistent behaviors that last different amounts of time. We propose an algorithm that automatically segments each speed time-series into subseries of appropriate durations, where for each subseries, the user has consistent and distinguishing behavior.

1.5 Key Contributions

In this paper, we make following four key contributions.

(1) We present the idea of using WiFi to identify users through their gestures, and implemented and evaluated a WiFi based system that identifies users through their gestures.
(2) We provide a measurement study to demonstrate that the hypotheses that different users have different but consistent behaviors of performing gestures, which has been shown to hold true in non-WiFi based settings, holds true in WiFi based settings as well.

(3) We propose a method to extract speed time-series from any given time-series of CSI measurements.
(4) We collected a comprehensive data set containing over 25,000 samples from 21 volunteers for 7 gestures in 4 different environments, and extensively evaluated the performance of WiID using this data set.

2 SPEED TIME-SERIES EXTRACTION

In this section, we describe how WiID processes any given gesture sample to extract the speed time-series from it. As a gesture sample is essentially a time-series of CSI measurements captured at a WiFi receiver when a user performs a gesture, we first give a quick overview of what CSI measurements represent.

2.1 Channel State Information (CSI)

WiFi devices typically consist of multiple transmit (Tx) and receive (Rx) antennas. An 802.11n/ac MIMO channel between each Tx-Rx antenna pair comprises multiple sub-carriers. Let $X(f,t)$ and $Y(f,t)$ be the frequency domain representations of transmitted and received signals, respectively, on an OFDM subcarrier with frequency f at time t between a given Tx-Rx pair. The two signals are related as $Y(f,t) = H(f,t) \times X(f,t)$, where $H(f,t)$ represents the complex-valued channel frequency response (CFR) for subcarrier with frequency f at time t. Let N_{Tx} and N_{Rx} represent the number of Tx and Rx antennas, respectively, and let S represent the number of subcarriers between each Tx-Rx pair. Each CSI measurement comprises $S \times N_{Tx} \times N_{Rx}$ CFR values, one for each subcarrier between each Tx-Rx pair. As WiFi network interface cards (NICs) generate CSI measurements repeatedly, we essentially obtain $S \times N_{Tx} \times N_{Rx}$ time-series of CFR values. Onward, we will call each time-series of CFR values a *CSI-stream*.

2.2 Denoising

WiID assumes that the WiFi based gesture recognition system, to which it is being added to, uses the PCA based method to remove noise from the CSI streams. The reason for requiring the PCA based denoising is its demonstrated robustness in removing various varieties of noises that are present in CSI streams [33–35]. While most prior gesture recognition systems already use the PCA based method for denoising, a few of them, such as WiGest [10], do not. For such systems, before proceeding with the next steps, WiID performs the PCA based denoising on the CSI streams itself using the method proposed in [35]. As demonstrated in [33, 35], among the principal components that result from the application of PCA, the third component contains the highest human movement signal to noise ratio. Therefore, WiID uses the third principal component for further processing. Onward, we will refer to this third component as the *denoised-stream*. As noise removal from CSI-streams is a well-studied topic in literature, we refer the interested readers to [33, 35].

2.3 Extraction

To extract the speed time-series from any given gesture sample, WiID first extracts the frequency time-series from it, and then converts it into the speed time-series. Recall that the frequencies in the denoised-stream appear due to the movement of the limb that performs the gesture, and the value of the frequency that appears in the denoised stream at any given time instant is directly proportional to the speed at which the limb was moving at that time instant. To clearly observe the frequencies introduced by the movement of the limb, WiID applies STFT on the time-series of the denoised-stream. STFT slides a window over the denoised stream, where at each sliding step, it applies fast fourier transform (FFT) on the values covered by the window in that step. The window size determines the tradeoff between frequency and time resolution of STFT. With a larger window size, STFT has higher frequency resolution but lower time resolutions, and vice versa. As our sampling rate is 1000 samples/sec, we chose a window size of 1024 samples and a window step size of 5 samples as it gives a good frequency resolution of about 1 Hz and a time resolution of 5 ms. An FFT at any given window step results in a vector of magnitudes of all frequencies in the portion of the denoised stream covered by the window in that step. As human movements give rise to frequencies less than 300Hz [34], we use only the first 300 values in this vector. We call this vector of length 300 a frequency vector.

A spectrogram of any given denoised stream is essentially a concatenation of all frequency vectors obtained from sliding the window over that denoised stream. Figure 2 shows an example spectrogram of a randomly selected gesture sample from our data set (we will describe our data sets in Section 3) as a heat map obtained using this approach. The hotter colors represent higher FFT amplitudes, which can be used to determine which frequencies were present in the denoised-stream at what time instant. Note from Figure 2 that the spectrogram does not show a crisp single frequency in each frequency vector, rather there is a band of high amplitude frequencies. This happens due to two reasons: 1) when performing any movement, different portions of a limb can move at somewhat different speeds, and thus, the reflections from those different portions introduce different but closely spaced frequencies in the denoised stream; 2) the distortions from any left over noise.

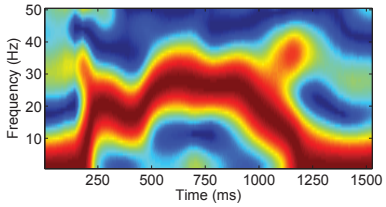


Fig. 2. Spectrogram of a sample

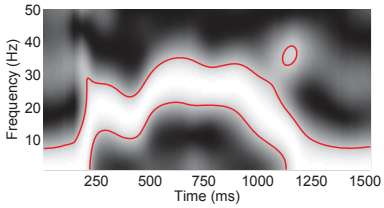


Fig. 3. Contour of high amplitude freqs.

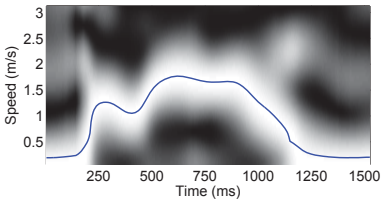


Fig. 4. Speed time-series of Fig. 2

The magnitudes of the frequencies that appear in the denoised stream due to human movements are much higher compared to the magnitudes of the remaining frequencies because the magnitudes of the remaining frequencies are only due to noise, which has already been largely removed. Therefore, to extract the portion of the spectrogram that represents high magnitudes, WiID creates contours such that all values inside the contours lie above the 90^{th} percentile of the magnitudes. Figure 3 shows the contours obtained from the spectrogram in Figure 2. Next, WiID sequentially processes each frequency vector in the spectrogram and selects the frequency value in the middle of the upper and lower points where the contour intersects with that frequency vector. This frequency value is an approximation of the frequency introduced by the movement of the limb at the time duration over which the frequency vector was calculated. Note that it is possible for some frequency vectors to intersect with multiple contours, such as the frequency vector at about 1180ms mark in Figure 3. In such a case, WiID selects the value in the middle of the upper and lower points of that contour that has the largest gap between the upper and lower points because the contours with smaller heights mostly result due to any left over noise or due to minor movements by either other users in the vicinity or other body parts of the same user.

After selecting a frequency value from each frequency vector, WiID has obtained the desired frequency time-series. It converts this frequency time-series into speed time-series by simply using the equation $v = f\lambda/2$, where λ is the wavelength of the WiFi signal, and f and v represent the values in the frequency and speed time-series, respectively. The motivation behind dividing the right hand side of this equation by 2 is that when a user moves his/her limb by a certain distance, the round trip length of the path reflecting from the limb increases by twice that distance [34, 35]. The speed time-series obtained from Figures 2 and 3 is shown in Figure 4.

3 DATA COLLECTION AND ANALYSIS

In this section, we first describe how we collected gesture samples from our volunteers in a variety of scenarios and present some statistics about our data sets. After that, we use the gesture samples in these data sets to study the hypothesis that different users have different behaviors of performing any given gesture but the behavior of any given user of performing the given gesture stays consistent over time.

3.1 Data Collection

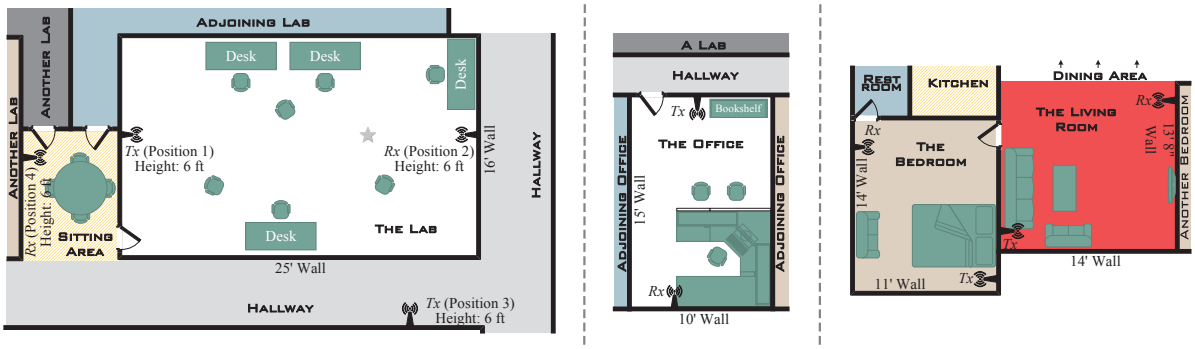
We collected two sets of gesture samples from our volunteers. We will refer to the first set of samples as data set 1 and the second set of samples as data set 2. We will use the samples in data set 1 to show that different users have different behaviors of performing any given gesture but any given user exhibits the same behavior every

time he/she performs the gesture. We will use the samples in data set 2 to show that the behavior of any given user of performing any given gesture stays consistent over time. Next, we first describe our data collection setup. After that, we describe how we collected the samples contained in the two data sets. Last, we describe which components were controlled and which were uncontrolled during the collection of these two data sets.

3.1.1 Data Collection Setup. We used the tool presented in [20] to acquire CSI measurements in the 2.4GHz band from an Intel 5300 WiFi NIC connected with 3 omnidirectional antennas. We used NETGEAR R6700 access point (AP) using 3 omnidirectional antennas and pinged it every 1ms to get a sampling rate of 1000 samples/sec. We collected these samples in four environments: a 400 ft^2 lab, a 150 ft^2 office, a 192 ft^2 living room, and a 154 ft^2 bedroom. Figure 5 shows the layouts of the four environments. When collecting samples in any given room, both WiFi receiver and WiFi transmitter were placed on the walls inside that room at a height of 6 ft from the floor. For the lab environment, the transmitter and receiver were placed at positions 1 and 2, respectively, as shown in Figure 5(a). The placement at positions 3 and 4 will be discussed later in the evaluation section.

For the lab, as shown in Figure 5(a), two sides are adjoined by hallways, one side by a common sitting area, and one side by a lab. The hallways are frequently used by students when going to their classrooms or to different floors, as the stairs leading to the upper and lower floors are accessed through these hallways. The common sitting area has chairs and a desk where students often sit and work on course projects. The lab adjacent to our lab is occupied by graduate students working on their research projects. Thus, all four structures adjoining the lab generate plenty of background movements outside the lab. For the office, as shown in Figure 5(b), two sides are adjoined by other offices, each occupied by a single individual for most of the time, one side by a hallway, and one by open space. The hallway is primarily used by graduate students and faculty. While there is still movement of people in the hallway and the offices, the amount of movement is less compared to the amount of movement around the lab. The living room and bedroom, shown in Figure 5(c), are in an apartment that is situated in a relatively quiet neighborhood. Therefore, there were not a lot of background movements around the apartment.

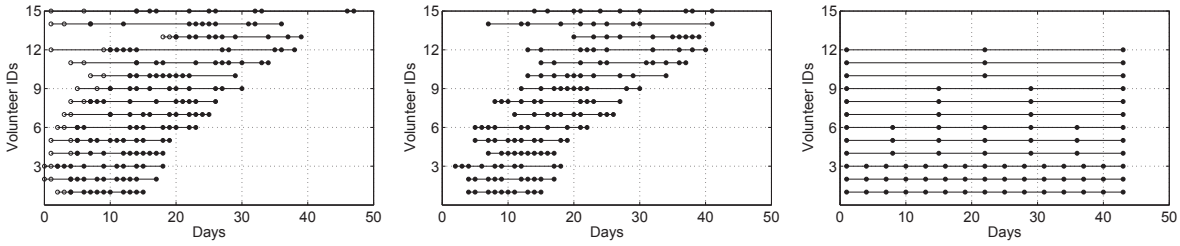
3.1.2 Data Set 1. In data set 1, we collected 21,000 samples from 15 volunteers for 7 gestures in the 4 environments. The names of the seven gestures and their abbreviations that we will use throughout the paper are: push and pull arm (PP), circular arm motion (CA), waving arm motion (WA), kicking (KK), open and close door (OC), raising and lowering both arms (RL), and extending both arms forward, and then moving to sides (EM). The 15 volunteers comprised 9 males and 6 females and their ages ranged from 20 to 37 years. We collected these samples after obtaining IRB approval and over a period of 47 days. To collect samples from each volunteer, we verbally described each gesture to the volunteer and asked the volunteer to perform the gesture to the best of his/her understanding. We collected 50 samples of each gesture from each volunteer in each environment and manually recorded the start and end times of each sample.



(a) Lab Environment (b) Office Environment (c) Bedroom and Living room Environments

Fig. 5. Layouts of the data collection environments (almost but not perfectly at scale)

If a volunteer is asked to perform the same gesture a large number of times over a short time span, he/she may either develop a temporary behavior during that time span that he/she can't replicate later, or may loose his/her behavior altogether in that time span due to fatigue. To avoid such gesture fatigue, we conducted multiple data collection sessions with each volunteer, where in each data collection session with any volunteer, we collected only 5 samples of each gesture and separated successive sessions by at least 24 hours. Before the first session with any volunteer, we also held two "try" sessions with that volunteer, where we asked the volunteer to simply practice each gesture multiple times, so that when we start the first data collection session, the volunteer is comfortable with those gestures. The two try sessions and the first data collection session were all also separated by at least 24 hour periods. We conducted the sessions in the lab and office environment on the same days due to the proximity of the two environments. Similarly, we conducted the sessions in the living room and bedroom on the same days due to the proximity of the two environments, but on different days from the sessions in the lab and office environments. Figure 6(a) shows a timing diagram visualizing the gap between the two try and ten data collection sessions of each volunteer in the lab and office environments. Figure 6(b) visualizes the gap between the ten data collection sessions of each volunteer in the living room and bedroom environments. We observe from these figures that the quickest data collection spanned over 14 days while the longest spanned 47 days.



(a) Data set 1: Lab and office (b) Data set 1: Living and bedroom (c) Data set 2: Lab

Fig. 6. Timing diagrams showing the days on which gesture samples were collected from different volunteers

3.1.3 Data Set 2. In data set 2, we collected 3915 samples from 12 volunteers (6 were the same as in data set 1) for 3 gestures (PP, CA, and EM) in the lab environment. The reason behind collecting samples in 1 environment instead of all 4 is that the purpose of data set 2 was to only study whether the behavior of users in performing gestures stays consistent over time; and for that, we do not need to study gestures in all 4 environments. The reason behind collecting samples for only 3 gestures instead of all 7 is also the same. The 12 volunteers comprised 8 males and 4 females with ages from 24 to 33 years. We collected these samples over a period of another 43 days.

We divided these 12 volunteers into four groups, each group comprising three volunteers. From the 3 volunteers in the first group, we collected samples every 3rd day over the period of 43 days. Thus, these 3 volunteers provided gesture samples on 15 different days over this period of 43 days, *i.e.*, on days 1, 4, 7, ... , 43. From the 3 volunteers in the second group, we collected samples every 7th day, *i.e.*, on days 1, 8, 15, 22, 29, 36, and 43. From the 3 volunteers in the third group, we collected samples every 14th day, *i.e.*, on days 1, 15, 29, and 43. Finally, from the 3 volunteers in the fourth group, we collected samples every 21st day, *i.e.*, on days 1, 22, and 43. Before the start of this 43 day period, just like in data set 1, we held two "try" sessions with each volunteer. Figure 6(c) shows a timing diagram visualizing the days on which we collected samples from different volunteers. On any day when any given volunteer was scheduled to provide gesture samples, we performed three data collection sessions with that volunteer, where the successive sessions on that day were separated by at least four hours. In each data collection session with any given volunteer, we collected 5 samples of each of the three gestures.

3.1.4 Additional Controlled and Uncontrolled Components during Data Collection. In each session with any given volunteer, we asked the volunteer to choose a random position of his/her choosing in the room, and provide gesture samples while standing there. We requested the volunteers to always face towards the WiFi receiver

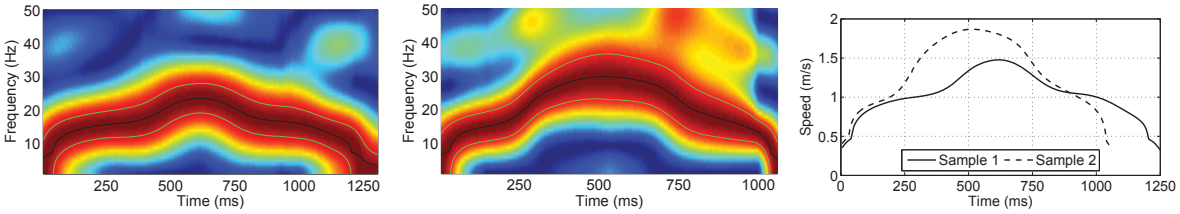
when providing gesture samples so that their orientation with respect to the receiver stays the same regardless of what position they choose. To incorporate the effects of static environmental changes, we randomly moved the furniture in the room each day before collecting samples on that day. When collecting gesture samples, we ensured that inside each room where we collected gesture samples, only one volunteer performed predefined gestures at any given time, and no other person moved while the volunteer performed the predefined gestures. While we ensured that *inside* each room no other person moved, the movements of the people *outside* each room were uncontrolled where an uncontrolled number of people performed their routine activities. As the walls of all four rooms where we collected gesture samples are standard drywalls, the movements of the people outside the rooms were not invisible to the Intel 5300 WiFi NIC, which we used to collect the CSI measurements.

3.2 Data Analysis

From the data sets described above, we made following four important observations. These observations are in keeping with some of the observations presented in prior studies, such as [28, 29].

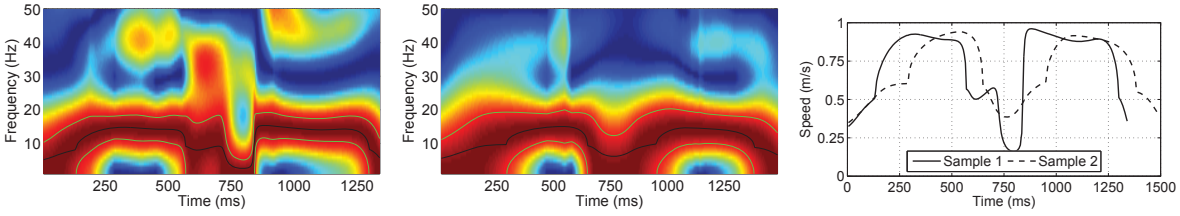
O1. *An individual user performs the same gesture with the same behavior.*
O2. *Different users perform the same gesture with different behaviors.*
O3. *The behavior of any user in performing the same gesture stays consistent over time if the user performs that gesture at least every week.*
O4. *If a user does not perform a gesture for more than two weeks, his/her behavior of performing that gesture for the first few times after the long pause does not match with his/her usual behavior; however, after performing that gesture a few times, the behavior returns to his/her usual consistent behavior.*

Figures 7(a) and 7(b) show the spectrograms of two samples of waving arm (WA) gesture from a volunteer. Figure 7(c) shows the speed time-series obtained from these two spectrograms using the method described in Section 2.3. Figures 8(a), 8(b), and 8(c) show the spectrograms and corresponding speed time-series, respectively, of two samples of the same WA gesture from another volunteer. We observe from these figures that the samples of a given gesture from the same user are very similar and at the same time quite different from the samples of that gesture from another user. This happens because for any given gesture, different users move their limbs at different speeds at different portions of that gesture. We made similar observations from the remaining samples.



(a) Spectrogram of sample 1 (b) Spectrogram of sample 2 (c) Speed time-series of samples 1 & 2

Fig. 7. Spectrograms and speed time-series of two randomly chosen samples of waving arm (WA) gesture of volunteer 3



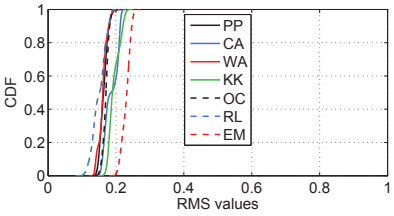
(a) Spectrogram of sample 1 (b) Spectrogram of sample 2 (c) Speed time-series of samples 1 & 2

Fig. 8. Spectrograms and speed time-series of two randomly chosen samples of waving arm (WA) gesture of volunteer 7

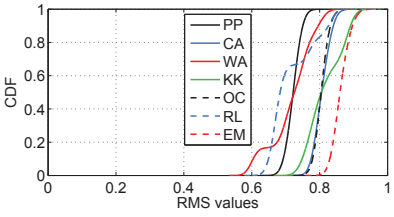
The description above was a qualitative observation from 4 randomly chosen samples from two volunteers. Next, we present a quantitative analysis over our entire two data sets to demonstrate the four observations. For the quantitative analysis, we first need to define a metric to quantify the similarity between two speed time-series obtained from any given pair of gesture samples. For this, we adapt the metric that was proposed in [28]. Let us represent one time-series with $\mathbf{T}_1$ and the other with $\mathbf{T}_2$, and let m_1 and m_2 represent the number of values in $\mathbf{T}_1$ and $\mathbf{T}_2$, respectively. The metric that we will use is the root mean squared (RMS) value of the time-series obtained by subtracting the normalized values of $\mathbf{T}_1$ from the corresponding normalized values of $\mathbf{T}_2$. To normalize the values in any given time-series $\mathbf{T}_i$, we employ the simple unity-based normalization [8], which brings all its values in the range of [0, 1]. More specifically, let $t_i[q]$ represent the q^{th} value in $\mathbf{T}_i$, where $1 \leq q \leq m_i$, and let $t_i^{\max}$ and $t_i^{\min}$ represent the maximum and minimum values in $\mathbf{T}_i$. The normalized value corresponding to any value $t_i[q]$ in the time-series $\mathbf{T}_i$ is calculated as $(t_i[q] - t_i^{\min})/(t_i^{\max} - t_i^{\min})$. To subtract one time-series from the other, the number of values in the two need to be equal; however, this often is not the case. Without the loss of generality, lets assume that $m_1 \leq m_2$. Before subtracting, we re-sample $\mathbf{T}_2$ at a sampling rate of m_1/m_2 to make $\mathbf{T}_2$ and $\mathbf{T}_1$ equal in terms of the number of values. Let $\mathbf{T}_R$ represent the time-series resulting after subtracting re-sampled $\mathbf{T}_2$ from $\mathbf{T}_1$. The RMS value of $\mathbf{T}_R$ is calculated as $\sqrt{\frac{1}{m_1}\sum_{q=1}^{m_1}(t_R[q])^2}$. Note that the RMS value of $\mathbf{T}_R$ always lies in the range of $[0, 1]$. An RMS value closer to 0 implies that the two time-series $\mathbf{T}_1$ and $\mathbf{T}_2$ are highly alike, while an RMS value closer to 1 implies that the two time-series are very different. For example, the RMS value between the two time-series in Figure 7(c) is 0.146 and that between the two time-series in Figure 8(c) is 0.172, whereas the RMS value between a time-series in Figure 7(c) and another in Figure 8(c) is 0.717.

3.2.1 Quantitative Analysis of Data Set 1. From the quantitative analysis of data set 1, we made the observations O1 and O2, which we demonstrate next. Figure 9(a) shows 7 cumulative distribution function (CDF) plots, one plot per gesture, obtained using the samples in data set 1 collected in the lab environment. A CDF plot corresponding to any given gesture is obtained using RMS values calculated from all pairs of samples of that gesture such that the two samples in any given pair were performed by the *same* volunteer. More specifically, recall that in data set 1, each of our 15 volunteers provided 50 samples for each gesture in the lab environment. To obtain the CDF plot for any given gesture, corresponding to each of the 15 volunteers, we calculated the RMS values between all C_2^{50} pairs of the samples of that gesture from that volunteer. As we have 15 volunteers, we get $15 \times C_2^{50} = 18375$ RMS values for that gesture. The CDF plot corresponding to any given gesture in Figure 9(a) is the CDF plot of the 18375 RMS values calculated for that gesture. We observe that the CDF plots for all 7 gestures are situated around RMS values of 0.1 to 0.2, which are close to the minimum RMS value of 0. This demonstrates the observation O1 that the behavior of any volunteer across multiple performances of any given gesture was very consistent.

Figure 9(b) also shows 7 CDF plots, one plot per gesture, also obtained using the samples in data set 1 collected in the lab environment. A CDF plot corresponding to any given gesture in this figure is obtained using RMS values calculated from all pairs of samples of that gesture such that the two samples in any given pair were performed by *different* volunteer. To obtain the CDF plot for any given gesture in this figure, corresponding to each of the 15 volunteers, we calculated the RMS values between each sample of the given gesture from that volunteer and each sample of the given gesture from every other volunteer. As each volunteer provided 50 samples of that gesture in the lab environment in data set 1, from any given pair of volunteers, we obtained 50^2 RMS values. As we can arrange the 15 volunteers in C_2^{15} pairs, we get a total of $C_2^{15} \times 50^2 = 262500$ RMS



(a) Between samples from same users

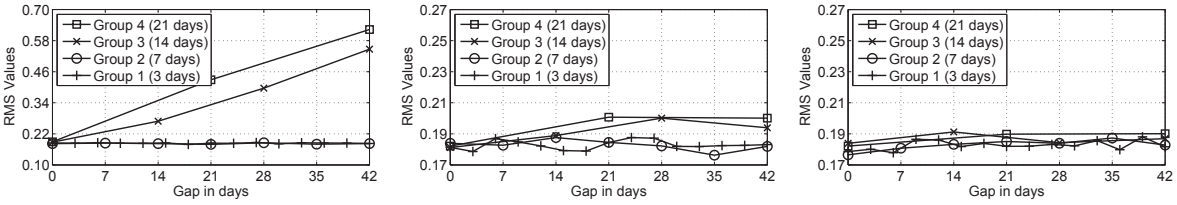


(b) Between samples from different users

Fig. 9. CDF of RMS values

values for that gesture. The CDF plot corresponding to any given gesture in Figure 9(b) is basically the CDF plot of the 262500 RMS values calculated for that gesture. We observe from this figure that the CDF plots for all 7 gestures are situated at RMS values greater than 0.6, *i.e.*, much farther away from the minimum RMS value of 0 compared to the CDF plots in Figure 9(a). This demonstrates the observation O2 that the behaviors of different volunteers in performing the same gesture were very different. We made the same observations from the samples collected in the office, living room, and bedroom environments.

3.2.2 Quantitative Analysis of Data Set 2. From the quantitative analysis of data set 2, we made the observations O3 and O4 listed at the start of Section 3.2, which we demonstrate next. Figures 10(a), 10(b), and 10(c) plot the consistency in the behaviors of the volunteers in the four groups over the period of 43 days. Each data point in Figure 10(a) corresponding to any given group at the x-axis value of d (where $0 \leq d \leq 42$) represents the average of the RMS values calculated from all unique pairs of samples in data set 2 such that the two samples in any given pair satisfy the following five conditions: 1) the two samples were collected during the first sessions on two different days; 2) the two sessions were separated in time by d days; 3) the two samples were of the same gesture; 4) the two samples were performed by the same volunteer; and 5) the volunteer who performed the two samples belonged to the given group. Data points in Figure 10(b) were obtained in the same way except that the first of the five conditions was that the two samples in any given pair were collected during the *second* sessions on two different days. Similarly, for the data points in Figure 10(c), the first of the five condition was that the two samples in any given pair were collected during the *third* sessions on two different days.



(a) First Sessions (b) Second Sessions (c) Third Sessions

Fig. 10. Consistency in the behaviors of the volunteers in the four groups over the 43 day period

From Figures 10(a), 10(b), and 10(c), we observe that the data points corresponding to the first group, *i.e.* the group whose 3 volunteers performed gestures every 3rd day, have low and stable values regardless of the value of the x-axis. Recall from the second of the five conditions that the x-axis represents the separation in days between pairs of sessions from which pairs of samples were selected to calculate the average RMS value. Similarly, we observe that the data points corresponding to the second group also have low and stable values regardless of the value of the x-axis. The stable values for the first and the second groups in each of the three figures demonstrate the observation O3 that if a user performs a gesture at least once every 7 days, his/her behavior of performing that gesture stays consistent over time. From Figure 10(a), we observe that the data points corresponding to the third group have higher values compared to the data points corresponding to the first and the second group, and show an increasing trend. The data points corresponding to the fourth group show a similar trend as the third group. These observations indicate that the similarity in behavior across pairs of samples of any given gesture by any given volunteer decreases remarkably as the volunteers take long pauses of 14 days or more, and the extent of similarity decreases with the increase in the duration of pause. However, we make a different observation from Figures 10(b) and 10(c): the data points corresponding to the third and the fourth group have low and stable values regardless of the value at the x-axis and the values are comparable to the values corresponding to the first and the second groups (notice that the y-axis range in Figures 10(b) and 10(c) is different from that in Figure 10(a)). Recall that Figures 10(a), 10(b), and 10(c) were obtained from samples collected in first, second, and third sessions, respectively. The low and stable values in Figures 10(b) and 10(c) for the third and the fourth group show that when using samples from sessions 2 and 3, users again demonstrated consistent behavior of performing

gestures over time, while the high and increasing values in Figure 10(a) showed exactly the opposite trend. These observations together demonstrate observation O4 that if a user does not perform a gesture for more than two weeks, his/her behavior of performing that gesture after the long pause becomes inconsistent (as indicated by values in Figure 10(a)), but this inconsistency lasts only for a few samples and the user's behavior returns to his/her usual behavior (as indicated by values in Figures 10(b) and 10(c)). In summary, the users indeed maintain consistent behavior over a long period of time, albeit, long pauses can introduce a temporary inconsistency.

4 FEATURE EXTRACTION

In this section, we describe how WiID extracts feature values from the training samples of any given gesture provided by any given user and how it selects appropriate features for generating classification models. Let G represent the number of gestures that the gesture recognition system recognizes, let U represent the number of users that have provided training samples for these G gestures, and let N_{ij} represent the number of samples of the i^{th} gesture provided by the j^{th} user, where $1 \leq i \leq G$ and $1 \leq j \leq U$. WiID uses the same method to extract and select features from the training samples of any given gesture provided by any given user. Thus, in the rest of this section, we describe this method considering the samples of an arbitrary gesture i from an arbitrary user j.

Before extracting the feature values from any given sample, WiID first converts that sample into a speed time-series using the method described in Section 2.3. To extract feature values, WiID needs to further segment each speed time-series into multiple smaller time-series because at different time-instants while performing a gesture, the human limb often has different speeds. Recall that we refer to these smaller time-series as subseries. If WiID did not segment the speed time-series into subseries, rather it extracted a single feature value from the entire speed time-series, it would essentially average out any distinguishing behaviors of users in the gesture samples, which would render it unable to distinguish across users.

Our goal is to segment any given speed time-series into subseries such that the average speed measured from each subseries characterizes the distinguishing behavior of the user who performed the gesture. There are three challenges to this goal. The first challenge is to determine how to segment the N_{ij} time-series into subseries, given an appropriate time duration as the segmentation guideline. The second challenge is to determine the time duration that is appropriate for use as the segmentation guideline. The third challenge is to determine which subseries to select whose average speed values should be included in the feature vector that WiID will use to generate classification models of users. Next, we present our solutions to these three technical challenges.

4.1 Time-series Segmentation

Given an appropriate time duration t_S as the segmentation guideline, WiID needs to segment each of the N_{ij} speed time-series into the same number of subseries so that from each time-series, it obtains the same number of features. However, as different speed time-series span different durations, segmenting each time-series into subseries of duration t_S will not always result in the same number of subseries. To address this issue, for each of the N_{ij} speed time-series, we first calculate $\lceil T_k/t_S \rceil$, where T_k is the time duration of the k^{th} speed time-series and $1 \leq k \leq N_{ij}$. From the resulting N_{ij} values, we select the most frequent value, denoted by S, as the number of subseries each speed time-series should be segmented into. Finally, we segment each of the N_{ij} speed time-series into S subseries such that all S subseries in any given speed time-series have equal durations. After segmenting time-series into subseries, we calculate a speed value from each subseries by averaging all values in it.

4.2 Segmentation Guideline

It is crucial to set the value of t_S correctly because if t_S is too small, the average speed values extracted from the corresponding subseries in the N_{ij} speed time-series may become inconsistent because when the values become instantaneous, they are unlikely to be consistent for the same user. Figure 11 shows the coefficient-of-variation [3], cv, of the average speed values extracted from the first subseries of the speed time-series of all 50 samples of OC gesture in the living room environment performed by a randomly selected volunteer in our data set when we

vary the subseries time duration from 10ms to 100ms. We observe from this figure that when t_S is very small, *i.e.*, when the subseries time duration is too small, cv is too large to be usable, and decreases as we increase t_S.

If t_S is too large, the average speed values extracted from corresponding subseries in the samples of a given gesture even from *different* users may become similar because all unique dynamics of individual users get too averaged out to be distinguishable. For example, treating all the samples in data set 1 of the EM gesture performed by all our volunteers in the lab environment as if they were all performed by the same person, the dashed line in Figure 12 shows that when t_S is 100ms, over 75% of the S subseries have consistent speed values (*i.e.*, the N_{ij} speed values in each such subseries have $cv < 0.25$), which means that these subseries do not have any distinguishing power among different users. It is therefore challenging to choose an appropriate value of t_S such that the resulting subseries are long enough to capture consistent behavior of the same user and short enough to maintain distinguishability across users.

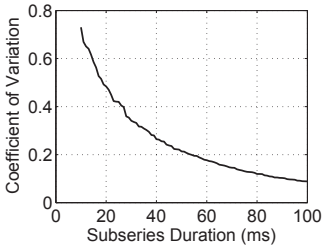


Fig. 11. Impact of subseries duration on cv

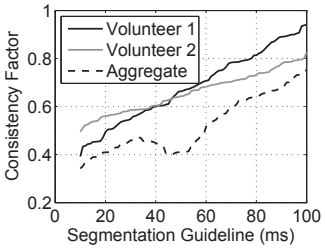


Fig. 12. Impact of t_S on consistency factor

Next, we describe how WiID determines the value of t_S that is appropriate for use as segmentation guideline. For this, we first define two metrics, namely *individual consistency factor* and *combined consistency factor*. Given the N_{ij} speed time-series of the i^{th} gesture from the j^{th} user, we call any subseries for which the N_{ij} speed values have $cv < 0.25$ a consistent subseries. When the number of subseries in each of the N_{ij} speed time-series, resulting from using t_S as segmentation guideline, is S, and the number of consistent subseries among these S subseries is C, the *individual consistency factor* of this set of N_{ij} samples is C/S. The *combined consistency factor* is the same as the individual consistency factor except that it is calculated using all $\sum_{j=1}^{U} N_{ij}$ samples of the i^{th} gesture collected from all U users. Figure 12 shows the combined consistency factor plot and two individual consistency factor plots for the RL gesture in the bedroom environment. We make two important observations from this figure. First, the individual consistency factors keep increasing as we increase the value of t_S. Second, the combined consistency factor has a significant dip when t_S is in the range from 35ms to 70ms. These two observations imply that when the segmentation guideline t_S lies in the range 35ms to 70ms, the durations of the subseries are long enough to capture consistent behavior of the same user and short enough to maintain distinguishability across users. We conducted the same measurement study with each of the 7 gestures collected in all 4 environments, and made similar observations. Thus, we choose t_S to lie in the range 35ms to 70ms.

4.3 Subseries Selection at Appropriate Resolutions

So far we have assumed that all subseries segmented from a time-series are obtained using a single value of t_S as the segmentation guideline. However, in reality, people have consistent and distinguishing behaviors that last for different durations at different portions of the gesture. Therefore, using a single value of t_S as the segmentation guideline is not appropriate. Next, we describe how WiID determines which portions of the given N_{ij} speed time-series should be segmented using what values of t_S.

We represent the entire duration of the time-series as a line with the initial color of white. Given the N_{ij} samples of the i^{th} gesture performed by the j^{th} user, WiID first segments the time-series using $t_S = 70$ms as the segmentation guideline. For each resulting subseries, it measures cv of the N_{ij} speed values extracted from that subseries from each of the N_{ij} samples. If $cv < 0.25$, then WiID selects this 70ms subseries for use in classification and colors the portion of the line corresponding to this subseries as black. After this round of segmentation, if any portion of the line is still left white, WiID moves to the next round of segmentation, this time using $t_S = 65$ms as the segmentation guideline. In this round, for each resulting subseries whose color is completely white, WiID again measures cv of the N_{ij} speed values, and if $cv < 0.25$, selects this 65ms subseries as well for

use in classification and colors the portion of the line corresponding to this subseries as black. WiID continues to repeat these segmentation rounds by decrementing the segmentation guideline t_S by 5ms in each round until either there is no white region left in the line or t_S has decremented to 35ms. If t_S has decremented to 35ms but there are still white regions, WiID does not use the speed values in the subseries spanning those white regions in generating classification models because the speed values in these white regions are not consistent across the N_{ij} samples. At this point, WiID has completed the segmentation of the N_{ij} speed time-series using appropriate time resolutions at different portions of the time-series, and has selected all subseries that have consistent and distinguishing average speed values for use in classification.

Note that WiID processes the N_{ij} training samples of any gesture j and user i without taking into account the samples of that same gesture from any other user. Consequently, the subseries that WiID selects in the samples of any given gesture from one user may be very different from the subseries that it selects in the samples of that gesture from another user. There are three reasons behind processing the samples of different users independently. First, it improves the accuracy of WiID by not missing any distinguishing behavior of any user. Second, it makes it straightforward to add new users or to remove existing users, as we will describe in the next section. Third, if a user that never provided training samples to WiID performs a gesture at runtime, this method enables WiID to easily determine that this user is not among any of the users enrolled with it.

5 CLASSIFIER TRAINING

In this section, we explain how WiID generates a classification model for each gesture of each user. As WiID selects different subseries (to calculate the average speed values from) in the samples from different users for any given gesture, a standard multi-class classification method cannot be used because that requires the average speed values to be calculated from the same subseries in the samples from different users. To overcome this challenge, for any given gesture i and user j, WiID generates an independent single class classification model using the average speed values calculated from the selected subseries in the N_{ij} samples of that gesture provided by that user. We call the set of these average speed values extracted from each sample a feature vector. Thus, WiID has N_{ij} feature vectors for gesture i of user j. To generate each single class classification model, we chose Support Vector Distribution Estimation (SVDE) with the Radial Basis Function (RBF) kernel due to their well-known effectiveness when the training data is only from one class (*i.e.*, the j^{th} user) while the test data can come from two classes (*i.e.*, the j^{th} user vs. all other users) [23, 27, 28]. We use the implementation of SVDE in libSVM [15].

To generate a classification model using SVDE, WiID first normalizes all N_{ij} values of each feature in the feature vector to come in the range $[0, 1]$; otherwise features with larger values dominate the classifier training and can negatively impact WiID's accuracy. SVDE has two tunable parameters: γ, a parameter for RBF kernel, and ν, a parameter for SVDE. WiID finds their optimal values by performing grid search on them along with 10-fold cross validation and selecting that pair of values for γ and ν that results in highest accuracy of correctly identifying the j^{th} user as the j^{th} user. Finally, WiID trains an SVDE classifier using the selected pair of values for γ and ν along with the N_{ij} feature vectors, and saves this classification model in a database for use at runtime.

Since there are G gestures and U users, WiID generates a total of $G \times U$ classification modes. As WiID generates a separate classification model of each user for each gesture, if a new user joins, all WiID has to do is to acquire training samples from the new user for all gestures, generate SVDE based classification models of that user, one for each gesture, using the method described above, and store them in the database. Similarly, if an existing user needs be removed, all WiID has to do is to not compare the test samples with the models of this user at runtime. Had we used the conventional multi-class classification methods, such as SVM, we would need to regenerate all models of all users for all gestures again every time a new user joined or an existing user left.

6 RUNTIME USER IDENTIFICATION

As soon as the WiFi based gesture recognition system detects and recognizes a gesture, WiID takes the denoised-stream captured during the start and end times of the detected gesture as input along with the decision made by

the gesture recognition system. Let this recognized gesture be any i^{th} gesture, where $1 \leq i \leq G$. Next, it converts this denoised-stream into speed time-series using the method described in Section 2.3, and evaluates it against the U classification models of this i^{th} gesture. Recall that U represents the number of user, and WiID generated a dedicated classification model for each of the U users for this gesture.

To evaluate the speed time-series of this recognized gesture against the classification model of the j^{th} user, where $1 \leq j \leq U$, WiID first calculates the average speed values from exactly the same subseries that it selected for the j^{th} user, as described in Section 4.3, and obtains a feature vector of the same length that it used in generating the classification model of the j^{th} user for the i^{th} gesture. Second, it scales the values in this feature vector using the same scaling factors that it used when scaling the values of the features in the N_{ij} feature vectors of the j^{th} user for this i^{th} gesture. Finally, it evaluates this normalized feature vector against the classification model of the j^{th} user for the i^{th} gesture, and calculates a likelihood value. Upon calculating a likelihood value from each of the U classification models for this gesture, WiID declares the gesture to have been performed by that user whose model returns the highest likelihood. If the likelihood value returned by each classification model is $< 50\%$, WiID declares that the gesture was not performed by any of the users enrolled with it, rather by someone unknown.

7 EVALUATION

We implemented WiID on a commodity PC with an 8 core Intel Xeon processor and 16GB RAM. Most of the evaluations that we present in this section were performed using data set 1. For evaluations that were performed using different data sets, we will first describe those data sets and then present the results. Next, we first present the overall accuracy of WiID. After that, we study the impact of various factors on the accuracy of WiID, which include number of users, number of gestures, heights and weights of users, clothing of users, placement of WiFi transmitter and receiver, orientation of users, and gesture fatigue. Last, we present the processing latency and memory consumption of WiID. In all the experiments that we present, we assume that *some* WiFi based gesture recognition system is in place that recognizes a gesture as soon the user performs it, and then feeds the denoised-streams between the start and end times of the recognized gesture to WiID to identify the user. To purely evaluate the accuracy of WiID, we assume that the WiFi based gesture recognition system to which WiID is being added to as an add-on module recognizes the gestures without any errors. An incorrect recognition of a gesture by the gesture recognition system will highly likely lead to an incorrect identification of the user by WiID, but such erroneous user identification is not the fault of WiID. WiID can identify the user correctly only when the gesture recognition system recognizes the gesture correctly.

7.1 Overall Accuracy

We study the overall accuracy of WiID through two sets of experiments on samples in data set 1. First, we train and test WiID using samples from the same environments. After that, we test WiID using samples from the environment that is different from the environment(s) from where the training samples were collected.

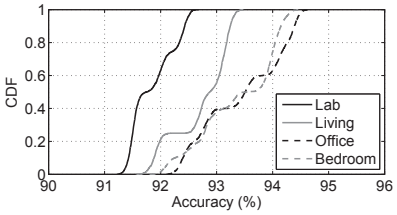
7.1.1 Accuracy in Same Environments. Recall from Section 3.1 that we collected samples from our volunteers in four different environments. To evaluate the overall accuracy of WiID in these four environments, we use two approaches: 1) cross validation, and 2) training and testing on samples collected on different days.

Cross Validation. To calculate the overall accuracy of WiID in each of these four environments using cross validation, we randomly selected $U = 5$ volunteers out of our 15 volunteers, took the 1750 samples (5 volunteers $\times$ 7 gestures $\times$ 50 samples per gesture per volunteer = 1750) that they provided in that environment, and performed k-fold cross validation, where $k = 10$ in our implementation. We used $U = 5$ because in most environments that are owned and shared by users, such as households or a shared office room, the number of occupants is usually around 5. We will study the impact of the changes in the value of U on the accuracy of WiID in the next section.

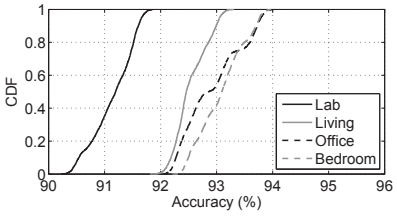
As per the principal of cross validation, in each fold, we select $(1/k) \times 50 = 5$ samples (which were not selected in any previous folds) of each volunteer for each gesture for testing, and the remaining $(1 - 1/k) \times 50 = 45$ for training. In other words, in each fold, we used 1575 samples (5 volunteers $\times$ 7 gestures $\times$ 45 samples per gesture

per volunteer = 1575) for training, and the remaining 175 for testing. At the end of all k folds, each of the 350 samples (7 gestures × 50 samples per gesture = 350) of each volunteer has been tested once. We calculate the overall accuracy of WiID for each of the five randomly selected volunteers as the percentage of his/her 350 samples from which WiID correctly identified him/her. Thus, at the end of this 10-fold cross validation, we get 5 accuracy values, one for each volunteer. We repeated this cross validation 500 times, each time with a different set of 5 randomly selected volunteers. This way, we obtained $5 \times 500 = 2500$ accuracy values for each environment.

Figure 13(a) shows four CDF plots, one for each of the four environments. The CDF plot for each environment is made from the 2500 accuracy values that we obtained for that environment. We observe from this figure that WiID achieves average accuracies of 91.8%, 93.4%, 92.7%, and 93.4% in the lab, office, living room, and bed room environments, respectively. The very similar accuracies in the four environments show that WiID is effective in identifying users irrespective of the environment they are in. We also observe from this figure that the CDF plots in all four environments have sharp slopes, which shows that not only the average accuracy of WiID is high, the standard deviation in the accuracy values is low, which indicates that WiID is accurate in identifying most users. Note that in evaluating WiID in each environment, the samples that we used for training and testing were collected at various different positions in that environment, and the furniture was randomly moved each day. Consequently, the CDFs in Figure 13(a) show the accuracy of WiID while incorporating the changes in user locations as well as static changes in the environment.



(a) Through cross validation



(b) Through training & testing on diff. days

Fig. 13. CDFs of WiID's overall accuracy

Training & Testing on Different Days. The motivation behind evaluating the accuracy of WiID using training and testing samples from different days is that in real-world, the users will not provide training samples on each day WiID is used, rather they will provide training samples only on the first few days when setting up WiID, and then on subsequent days, WiID should be able to identify them at runtime. To calculate the overall accuracy of WiID in each of the four environments using training and testing samples from different days, we randomly chose $U = 5$ volunteers out of our 15. Recall form Section 3.1.2 that in data set 1, each of the 10 data collection sessions with any given volunteer was conducted on a different day. Let us represent the i^{th} session with S_i, where $1 \leq i \leq 10$. For each value of i, we selected the 5 samples of each volunteer for each gesture from data collection session S_i for testing and the 45 samples of each volunteer for each gesture from the remaining 9 sessions for training. Thus, we had 10 rounds of training and testing corresponding to the 10 values of i, where in each round we had 1575 samples (5 volunteers × 7 gestures × 45 samples per gesture per volunteer = 1575) for training, and the remaining 175 for testing. At the end of all 10 rounds, each of the 350 samples (7 gestures × 50 samples per gesture = 350) of each volunteer has been tested once. We calculate accuracy of WiID for each of the five randomly selected volunteers as the percentage of his/her 350 samples from which WiID correctly identified him/her. Consequently, at the end of the 10 rounds, we get 5 accuracy values, one for each volunteer. We repeated this process 500 times, each time with a different set of 5 randomly selected volunteers. This way, we obtained $5 \times 500 = 2500$ accuracy values for each environment.

Similar to Figure 13(a), Figure 13(b), shows four CDF plots, one for each of the four environments. The CDF plot for each environment is made from the 2500 accuracy values that we obtained for that environment. We observe from this figure that WiID achieves average accuracies of 91.1%, 92.9%, 92.5%, and 93.1% in the lab, office, living room, and bed room environments, respectively, with sharp slopes of the CDFs. On comparing the CDFs in Figure 13(b) with the CDFs in Figure 13(a), we observe that the accuracies when training and testing on samples from different days are slightly lower compared to the accuracies calculated through cross validation. This was expected as it is well-known that cross validation usually introduces a small over-estimation error in accuracy.

7.1.2 Accuracy across Different Environments. Next, we study the accuracy of WiID when the training and testing samples are collected from different environments. For this, we use two approaches: 1) training samples from multiple unseen environments, and 2) training samples from a single unseen environment.

Training Samples from Multiple Unseen Environments. For this set of experiments, we used gesture samples from three environments for training and the remaining one environment for testing, and repeated this for all 4 possible combinations, where in each combination, we used samples from a different environment for testing. Let E_j represent any arbitrary environment, where $1 \leq j \leq 4$ (because we have four environments in our data set 1). In any given combination out of the four combinations, where the samples from environment E_j are used for testing, we randomly selected $U = 5$ volunteers out of our 15 volunteers and perform a round of classification where we uses the 1750 samples of the 5 volunteers (5 volunteers × 7 gestures × 50 samples per gesture per volunteer = 1750) collected in the the environment E_j as test samples and 5250 samples of the same 5 volunteers collected in the remaining three environments (3 remaining environments × 1750 samples per environment = 5250) for training. On completing this round of classification, each of the 350 samples (7 gestures × 50 samples per gesture = 350) of each of the five volunteers collected in the environment E_j has been tested once. We calculate accuracy of WiID for each of the five randomly selected volunteers as the percentage of his/her 350 samples from which WiID correctly identified him/her. Consequently, at the end of this round of classification, we get 5 accuracy values, one for each volunteer. We repeated such rounds of classification 500 times, each time with a different set of 5 randomly selected volunteers. This way, we obtained $5 \times 500 = 2500$ accuracy values for the environment E_j. We executed the entire process described above for all four values of j and obtained 2500 accuracy values for each environment.

Figure 14 shows four bars, one for each environment, representing the average of the 2500 accuracy values that we obtained for that environment. We observe from this figure that WiID achieves average accuracies of 92.2%, 94.3%, 93.1%, and 95.7% when using the lab (L), office (O), living room (R), and bed room (B) environments, respectively, as test environments. On comparing these average values from the average values obtained from Figures 13(a) and 13(b), we observe that WiID actually achieved higher accuracy when training samples came from unseen (but multiple) environments compared to when they came from the same environment. This increase in accuracy is due to two reasons. First, as we used training samples from three environments (albeit different from the test environment) to obtain the results in Figure 14, the number of training samples used in each round of classification were over three times more than the number of training samples used in each round of classification when obtaining the results in Figures 13(a) and 13(b). The larger number of training samples led to the development of more robust classification models, which in turn led to higher accuracy. Second, as the receiver was always placed at the height of 6 ft on the wall in each environment, there was always a line of sight path between the volunteer and the receiver. Consequently, the speed time-series of the same gesture from the same volunteer in the samples collected in different environments were quite similar to each other.

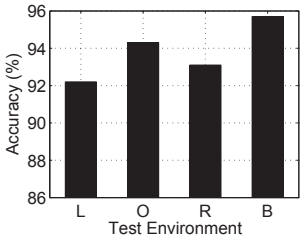


Fig. 14. WiID's accuracy with training samples from three unseen environments

Training Samples from Single Unseen Environment. For this set of experiments, we used gesture samples from one environment for training and the remaining three for testing, and repeated this for all 4 possible combinations, where in each combination, we used samples from a different environment for training. Let E_j again represent any arbitrary environment, where $1 \leq j \leq 4$. Different from above, where we used samples collected in environment E_j for testing, here we will use samples collected in E_j for training. More specifically, in any given combination out of the four combinations, where the samples from environment E_j are used for training, we randomly selected $U = 5$ volunteers out of our 15 and performed a round of classification, where we use the 1750 samples of the 5 volunteers (5 volunteers × 7 gestures × 50 samples per gesture per volunteer = 1750) collected in the environment E_j as training samples and 5250 samples of these 5 users collected in the remaining three environments (3 remaining

environments × 1750 samples per environment = 5250) for testing. On completing this round of classification, each of the 350 samples of each of the five volunteers collected in each of the three test environments has been tested once. We calculate the accuracy of WiID for each of the five volunteers in each test environment as the percentage of his/her 350 samples collected in that environment from which WiID correctly identified him/her. Thus, at the end of this classification round, we get 5 accuracy values (one per volunteer) for each of the three environments. We repeated the classification rounds 500 times, each time with a different set of 5 randomly selected volunteers. This way, we obtained $5 \times 500 = 2500$ accuracy values for each of the three test environments. We repeated the entire process described above three more times, each time using a different environment as the training environment. As each environment is used as training environment in one combination and test environment in the remaining three, we obtained $2500 \times 3 = 7500$ accuracy values for each environment.

Figure 15 shows four sets of four bars, one set per combination, where in each set, the bar corresponding to the environment that was used as the training environment is actually missing because the samples of the environment used for training were not used for testing. Each bar corresponding to any given environment in any given set of bars plots the average of the 2500 accuracy values obtained for that environment in the experiment from which the given set of bars was generated. We observe from this figure that when training samples were used from living room, bedroom, or office environments, the accuracy for the remaining environments was comparable to the accuracy values that we have being seeing until now. This is because there were very few background movements outside the office, living room, and bedroom environments when collecting gesture samples, as described in Section 3.1.1. However, when we used samples from the lab environment for training, the accuracy on the samples from the remaining three environments was relatively low, primarily because the lab environment was noisy compared to the remaining three. Based on these observations, we recommend that when collecting training samples at the time of setting up WiID, the users should select those times to provide training samples when the background movements are minimal.

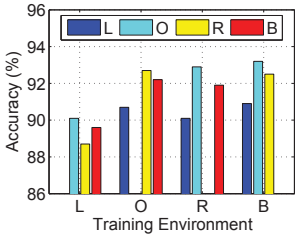


Fig. 15. WiID's accuracy with training samples from one unseen environments

7.2 Impact of the Number of Users

To study the impact of the number of users between which WiID should distinguish, we repeated the same experiments as described in Section 7.1.1 for eight different values of U, ranging from 3 to 10. Figure 16 shows eight sets of 4 bars, one bar for each environment. Each set of 4 bars corresponds to one value of U in the range [3,10]. Each bar in this figure corresponding to any value of U in any environment is the average of the $U \times 500$ accuracy values for that environment (just like in Section 7.1.1, where we used $U = 5$ and obtained $5 \times 500 = 2500$ accuracy values for each environment). As we can expect, the average accuracy of WiID reduces with the increase in the number of users between which it has to distinguish because with the increase in the number of users, as the number of classes increase (each user represents one class), the probability of confusion among the users increases. Nonetheless, we observe that the average accuracy of WiID is still above 90% for up to 7 users. Seven is a large enough number for most households and other shared spaces where multiple users may own and interact with shared smart devices. Even for 8, 9, and 10 users, the average accuracy of WiID is above 85%.

7.3 Impact of the Number of Gestures

To study the impact of the number of gestures, we repeated the same set of experiments as described in Section 7.1.1, this time keeping the value of the number of users U fixed at 5 and increasing the number of gestures G from 1 to 7. Figure 17 shows 7 sets of bars, one for each value of G, in the living room environment. We do not show figures for the remaining three environments due to space limitation. The observations we made from them are the same as those that we describe next. Each set of bars in Figure 17 contains $G + 1$ bars. In each set, the left most

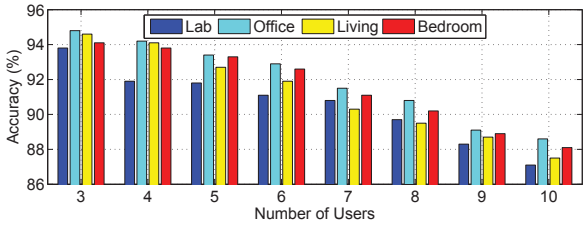


Fig. 16. Impact of the number of users on WiID's accuracy

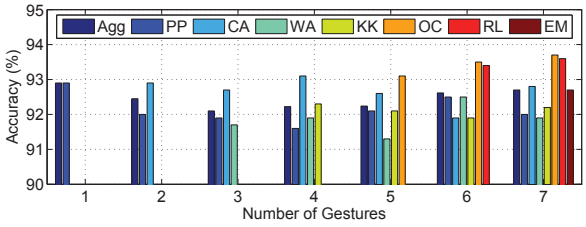


Fig. 17. Impact of number of gestures on WiID's accuracy

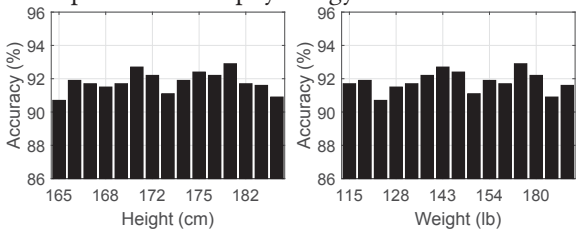
bar represents the average accuracy of WiID in identifying the users across all gestures and all U users, whereas each of the remaining G bars represent the accuracy of WiID in identifying users from one specific gesture. We observe from this figure that the average accuracy of WiID in identifying users stays largely unaffected by the number of gestures. This happens because recognizing the gesture is not the responsibility of WiID but of the gesture recognition system that is being augmented with WiID. WiID's responsibility is to identify the individual performing the gesture once it is provided with the correctly recognized gesture. Consequently, as far as the accuracy of WiID is concerned, it does not get significantly affected by the number of different gestures being recognized by the gesture recognition system.

7.4 Impact of User Heights and Weights

The motivation behind studying the impact of user heights and weights on the accuracy of WiID is to see whether WiID's accuracy is impacted by any physiological properties of human body. Figures 18(a) and 18(b) plot the accuracies of WiID for users sorted with respect to their heights and weights, respectively. Each bar in the two figures for a user with any given height/weight shows the average of 500 accuracy values calculated for that user in the lab environment. More specifically, we randomly picked $U = 5$ volunteers from our dataset, performed 10 fold cross validation on their samples, as explained in Section 7.1.1, and obtained an accuracy value for each volunteer from this experiment. We repeated this experiment 1500 times, each time selecting a different set of 5 volunteers such that each volunteer was selected in exactly 500 runs of the experiment out of the 1500 runs.

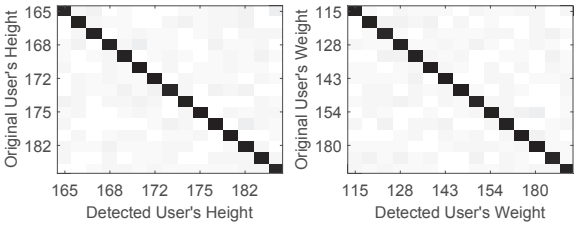
We observe from these two figures that their is no apparent trend in WiID's accuracy with changes in volunteers' heights and weights. This shows that the physiology of the user does not have any significant impact on the accuracy of WiID. We made the same observation from the data collected in the office, bedroom, and living room environments. Due to space limitation, we do not show the figures for those three environments here.

Figures 19(a) and 19(b) show the confusion matrices where the volunteers are arranged in the ascending orders of their heights and weights. The misclassifications have been enhanced (*i.e.*, intentionally darkened) in these figures to make any trends standout. However, we do not observe any such trends from these figures that would indicate that WiID confuses users that have similar heights and/or weights. Thus, we can conclude that the classification errors that WiID incurs are only statistical in nature and are randomly distributed. WiID's accuracy is independent of the physiology of the users.



(a) Height (b) Weight

Fig. 18. Impact of users' heights and weights on WiID's accuracy



(a) Height (b) Weight

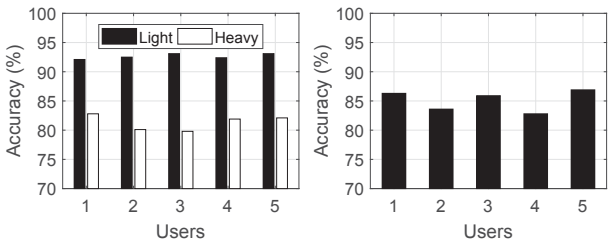
Fig. 19. Confusion matrices of WiID with respect to users' heights and weights

7.5 Impact of Clothing

To study how clothing may impact the accuracy of WiID in identifying users, we did 3 more data collection sessions with five of our volunteers in the living room environment. In each of the three sessions with any volunteer, we asked the volunteer to wear a different number of layers of clothes and provide 10 samples for each of the 7 gestures. In the first session, the volunteers wore only a single layer of clothes, mostly a shirt; in the second session, the volunteers wore another layer of winter clothes, such as sweaters or blazers; and in the third, the volunteers wore heavy winter clothes, such as multiple layers of sweaters as well as gloves and neck scarves.

Using this data, we first trained WiID using the samples collected in the first session, and then evaluated the samples collected in the second and the third sessions. Figure 20(a) plots five pairs of bars, one pair for each volunteer. The left bar in the pair for any given volunteer shows the percentage of samples of that volunteer collected in session 2 from which WiID identified him/her correctly. Similarly, the right bar represents the percentage of samples of that volunteer collected in session 3 from which WiID identified him/her correctly. We observe from this figure that the accuracy of WiID in session 2 is the same as the average accuracies of WiID reported in Section 7.1.1. This shows that light winter clothing does not impact the accuracy of WiID. However, we observe that with very heavy winter clothing, the accuracy of WiID drops by an average of about 11%. This shows that heavy winter clothing makes people loose their usual behavior. Fortunately, the shared environments where WiID is required are indoor environments, and most smart environments these days are temperature controlled. Consequently, the users do not wear heavy winter clothing in such smart environments, and this degradation of performance of WiID with heavy winter clothing will not impact WiID in most real-world settings.

We also studied whether users even have a consistent behavior if both training and testing was done using the samples collected in the third scenario. Figure 20(b) plots the average 10-fold cross validation accuracy for each of the 5 volunteers where the training and testing was done on the samples collected in the third session. We observe that the accuracy is about 7% lower compared to the accuracies reported in Section 7.1.1. This shows that with heavy winter clothing, it just becomes difficult for users to have a consistent behavior across samples.



(a) Trained on regular, tested on light and heavy winter clothing (b) Trained and tested on heavy winter clothing

Fig. 20. Impact of clothing on the accuracy of WiID

7.6 Impact of the Placement of WiFi Transmitter and Receiver

In this section, we study the impact of the placement of WiFi transmitter and receiver on the accuracy of WiID. Recall from Section 3.1 that when collecting samples for data set 1 in the lab environment, both WiFi transmitter and WiFi receiver were placed inside the lab (positions 1 and 2 respectively), as shown in the Figure 5(a). To study the impact of the placement of the WiFi transmitter and/or the WiFi receiver outside the room, we performed two new rounds of data collection. For this evaluation, we chose the lab environment as it presents the most challenging environment for WiID among the four environments as seen in Section 7.1. In the two new rounds of data collection, we collected samples for three gestures (PP, CA, and EM) from 5 volunteers (three males and two females). These 5 volunteers were a subset of the volunteers that provided us samples for data set 1. In the first round of data collection, we kept the WiFi receiver at position 2 and placed the transmitter outside the lab across the hallway at position 3, as shown in Figure 5(a), and performed three data collection sessions with each volunteer, where the successive sessions were separated by at least four hours. In each data collection session with any given volunteer, we collected 5 samples of each of the three gestures from that volunteer. This way, we collected 225 samples in this round of data collection (5 volunteers × 3 gestures × 3 sessions × 5 samples per

volunteer per gesture per session = 225). In the second round of data collection, we placed both the transmitter and the receiver outside the lab: transmitter across the hallway at position 3 and receiver across the sitting area at position 4, as shown in Figure 5(a), and collected another 225 samples. In this round, we ensured that there were no people moving in the sitting area. In the third round of data collection, we kept both the transmitter and the receiver at positions 3 and 4, respectively, and collected another 225 samples, this time with people moving in the sitting area.

Recall that for each of our 5 volunteers and for each of the three gestures (PP, CA, and EM), in our data set 1, we have 50 samples per gesture collected in the lab environment. Thus, we use these $5 \times 50 \times 3 = 750$ samples from data set 1 for training and the three sets of the 225 samples collected in the three rounds of data collection described above for testing. Figure 21 shows three bars, where each bar represents the percentage of the 225 samples collected in the data collection round corresponding to that bar from which WiID identified the users correctly. WiID's accuracy for rounds 1, 2, and 3 turned out to be 90.7%, 88.4%, and 80.4%, respectively. Recall from Figure 13(b) that WiID achieved 91.1% accuracy when the transmitter and receiver were both places in the same room. The accuracy of 90.7% for the data collected in the first round in this section shows that placing the transmitter outside does not have any notable impact on the accuracy of WiID compared to when both transmitter and receiver are inside the lab. The reason being that it is the signal that reflects from the user and arrives at the receiver that is important, and the placement of the transmitter outside the lab did not affect the path of the signal from the user to the receiver (as long as the transmitter is not so far away that its signal becomes very weak by the time is arrives at the receiver). The accuracy of 88.4% for the data collected in the second round shows that even though the line of sight between the volunteers and the receiver is lost, the drop in accuracy is not significant (only 2.7%) because due to the lack of the movements of other people in the vicinity, the receiver is still able to capture the signal reflected from the user propagating through the drywall. However, when other people were present and moving in the sitting area, the accuracy of WiID dropped by 10.7%, as shown by the accuracy of 80.4% for the data collected in the third round. This large drop occurs because the weak signal reflected from the user inside the lab drowned under the more powerful signals reflected from the people moving in the sitting area. Recall that the movements of the people in the same sitting area did not have a significant impact when the receiver was placed inside the lab, as shown by the 91.1% accuracy in Figure 13(b). We conclude that while WiID achieves acceptable accuracy if the receiver is placed in an adjacent room and that adjacent room does not have significant movements while a user performs predefined gestures, we recommend that to achieve the best performance with WiID (1) the receiver should be in the same room where gesture recognition and user identification is desired, and (2) there should be a line of sight between the user and the receiver.

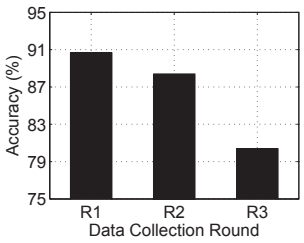


Fig. 21. Impact of the placement of WiFi transmitter and receiver on WiID's accuracy

7.7 Impact of User Orientation

Recall from Section 3.1.4 that even though our volunteers changed their positions across session, they always faced towards the WiFi receiver when providing gesture samples. Thus, their orientations with respect to the receiver always stayed the same. In this section, we study how changes in the orientations of users impact the accuracy of WiID. For this, we performed another round of data collection. Again, we chose the lab environment for the same reasons as mentioned in Section 7.6 and collected gesture samples for the same three gestures (PP, CA, and EM) from the same 5 volunteers as in Section 7.6. For this round of data collection, we kept both WiFi transmitter and receiver inside the room at positions 1 and 2, respectively, and performed three data collection sessions with each volunteer, where the successive sessions were separated by at least four hours. In each data collection session with any given volunteer, we asked the volunteer to stand at a fixed location in front of the

WiFi receiver at the position marked with star in Figure 5(a) and provide 5 samples of each of the three gestures while randomly changing his/her orientation before each sample. The motivation behind keeping the position fixed was so that we can purely evaluate the impact of changes in orientation. This way, we collected 225 samples in this round of data collection (5 volunteers × 3 gestures × 3 sessions × 5 samples = 225).

We use the 50 samples per gesture (collected in the lab environment) in our data set 1 for each of the 3 gestures from each of our 5 volunteers for training and the 225 samples collected in the round of data collection described just above for testing. WiID correctly identified users from 176 out of these 225 samples, achieving an accuracy of 78.2%, which is 12.9% lower compared to the 91.1% accuracy seen in Figure 13(b) that it achieved when volunteers did not change their orientations. This significant drop in accuracy occurred because when the orientation of a user changes, the speed time-series gets impacted, and WiID is unable to identify a user correctly anymore.

To overcome this problem, we used the translation function proposed by WiAG [33] which translates any given gesture sample obtained in any given orientation to any desired orientation. WiAG essentially makes WiFi based gesture recognition systems agnostic to user orientation and position. We augmented the translation function of WiAG with WiID, *i.e.*, before evaluating any gesture sample, we first apply the translation function to automatically rotate the gesture sample such that the rotated gesture sample is comprised of the same CSI values as the CSI values that would result from the user performing that gesture while facing towards the receiver, and then used WiID to identify the user from this automatically rotated sample. With the use of this translation function, WiID correctly identified users from 203 out of these 225 samples, achieving an accuracy of 90.2%, which is very close to the 91.1% accuracy seen in Figure 13(b). Thus, by augmenting the translation function of WiAG with WiID, WiID becomes fairly agnostic to user orientation.

7.8 Impact of Gesture Fatigue

Recall from Section 3.1 that to avoid gesture fatigue, in each data collection session with any volunteer, we collected only 5 samples of each gestures. In this section, we study how gesture fatigue impacts the accuracy of WiID. For this, we performed another round of data collection. Again, we chose the lab environment for the same reasons as mentioned in Section 7.6 and collected gesture samples from same 5 volunteers as in Section 7.6, but this time only for a single gesture CA. We did not collect samples for all seven gestures because with the CA gesture, the user would get just as much fatigued as with any other gesture. For this round of data collection, we again kept both WiFi transmitter and receiver inside the room, as shown in Figure 5(a), and performed three data collection sessions with each volunteer, where the successive sessions were separated by at least four hours. On day i, where $1 \leq i \leq 5$, during each data collection session with each volunteer, we asked the volunteer to perform the gesture $5 \times i$ times, taking a pause of about 10 seconds between successive samples. This way, on day 1, in each of the three sessions, each volunteer performed the gesture 5 times. Similarly, on days 2, 3, 4, and 5, each volunteer performed the gesture 10, 15, 20, and 25 times, respectively, in each session. Consequently, we obtained $75i$ samples on the ith day (5 volunteers × 3 sessions × $5i$ samples per session per volunteer = $75i$).

Recall that for each of our 5 volunteers, in data set 1, we have 50 samples of the CA gesture collected in the lab environment. We used these $5 \times 50 = 250$ samples from data set 1 for training and the $75i$ samples on the i^{th} day for testing, where $1 \leq i \leq 5$. Figure 22 plots the average accuracy of WiID on samples # j across all sessions, where $1 \leq j \leq 25$. For example, the average accuracy corresponding to sample # 2 in Figure 22 is calculated by evaluating the second sample in each of the three sessions on each of the five days (5 days × 3 sessions × 5 volunteers = 75 samples) and calculating the percentage of these 75 samples that WiID correctly recognized. Similarly, the average accuracy corresponding to sample # 7 in Figure 22 is calculated by evaluating the seventh sample in each of the three

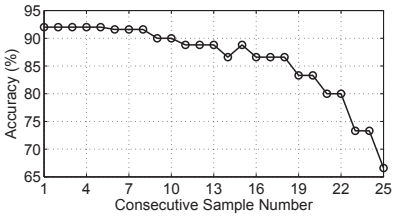


Fig. 22. Impact of gesture fatigue on WiID's accuracy

sessions on days 2, 3, 4, and 5 (recall that on day 1, each session only had 5 samples, so there is no sample # 7 in any of the three sessions on day 1; 4 days × 3 sessions × 5 volunteers = 60 samples) and calculating the percentage of these 60 samples that WiID correctly recognized. We observe from this figure that up until about 8^{th} consecutive performance of the gesture, the users do not experience significant gesture fatigue, and WiID achieved an accuracy above 90%. Beyond 8 performances, the accuracy starts to drop gradually, and the rate of drop becomes more pronounced after about 17 performances of the gesture. In real world, the user would hardly ever have to perform the same gesture more than 8 times consecutively over a short time span. If an application requires a user to perform a gesture even more frequently, then WiFi based gesture recognition and user identification is not a user friendly choice, and an alternative modality of user input should be considered.

7.9 Processing Latency and Memory Consumption

Next, we present the processing latency and memory consumption of our implementation of WiID on three different computers. We name these three computers C_1, C_2, and C_3, where C_1 is equipped with Intel Xeon E5-1620 v3 processor and 16GB RAM, C_2 is equipped with Intel i7-6700 processor and 8GB RAM, and C_3 is equipped with Intel i5-4300 processor and 4GB RAM. To generate a classification model of each user for any given gesture using 50 training samples, our implementation of WiID took an average of 13.1, 14.2, and 17.2 seconds on C_1, C_2, and C_3, respectively. While these execution times may seem long, these are practically not problematic because generation of classification models is a one time cost incurred at the time of setting up WiID, and does not impact WiID's runtime performance. When identifying a user from a given gesture sample, our implementation of WiID took an average of only 43.3, 47.1, and 59.2 milliseconds on C_1, C_2, and C_3, respectively, to evaluate the sample against the classification model of each user for that gesture. In other words, after the gesture recognition system recognizes a gesture sample, WiID takes approximately just $U \times 59.2$ milliseconds on our least powerful computer to identify the user from that gesture. Recall that U represents the number of users that WiID has to distinguish between. Thus, even when $U = 10$, which is already too large a value of U for most real world settings, WiID takes just a little over half a second to identify a user on our least powerful computer. This shows that WiID is amenable for use in realtime. The reason behind only a small increase in the processing times when going from C_1 to C_2 to C_3 is that our implementation of WiID utilizes only a single core of any given CPU and has not been optimized to benefit from multiple cores available on most modern processors. We also recorded the memory consumed by our implementation. When generating classification models, our implementation used an average of 1683MB, 1627MB, and 1611MB on C_1, C_2, and C_3, respectively. The maximum memory consumption on each of these computers stayed under 2GB. The memory consumption when evaluating a test sample against classification models was an order of magnitude smaller. This memory requirement is easily met on commodity computers these days.

8 DISCUSSION AND LIMITATIONS

Compatibility Requirements. To be compatible with a WiFi based gesture recognition system, WiID requires three things from it, as indicated by the three arrows going from the gray boxes to the green boxes in Figure 1. These three things include: 1) the name of the recognized gesture, 2) the denoised CSI values constituting the training samples, and 3) the denoised CSI values constituting the detected gesture. For the first of the three requirements, any WiFi based gesture recognition system always outputs the name of the recognized gesture (or an appropriate identifier assigned to the gesture) as that is the primary task of any gesture recognition system. Consequently, in terms of this requirement # 1, WiID is compatible with any WiFi based gesture recognition system. For the second of the three requirements, there are two scenarios where WiID may not be compatible with a given WiFi based gesture recognition system. The first scenario is where the gesture recognition system does not use CSI values, rather uses some other wireless channel metrics such as RSS, as in the case of WiGest [10]. The second scenario is where the gesture recognition system does not use training samples at all. Please note that currently, all existing WiFi based gesture recognition systems do require training samples for the gestures

that they have to recognize. For the third of the three requirements, just like for the second requirement, WiID may not be compatible with a given WiFi based gesture recognition system if the gesture recognition system does not use CSI values, rather uses some other metrics.

Implementation Optimization. Our implementation of WiID that we have used in this paper was only a proof of concept implementation to evaluate the accuracy of WiID in various scenarios, and was not a production quality implementation. For a production quality implementation, there is a significant room for improvement by parallelizing the executions of the various modules of WiID and benefitting from the multiple cores available on most CPUs these days. As an example of the possible improvements, when generating a classification model, the SVDE classification algorithm performs several independent iterations, which if performed in parallel, can speed up the execution of WiID by a factor of up to the number of parallelly executing streams. Similarly, when evaluating an unknown sample against the classification models of all users, if multiple cores are utilized to evaluate the test sample against multiple enrolled users in parallel, WiID's (already small) user identification time can also be cut by a factor of up to the number of parallelly executing streams.

Simultaneous and Background Movements. Currently, WiID assumes that at any given time, only a single user performs a predefined gesture. WiID further assumes that while a user performs a predefined gesture, there are no background movements inside the same room as the user. If multiple users perform predefined gestures simultaneously or if there are background movements inside the same room, WiID cannot identify the users from the gestures. WiID's accuracy, however, is not impacted significantly by the movements of people outside the room as demonstrated by our evaluations on the samples collected in the lab environment where there were a lot of routine background movements of people outside the lab. The authors of WiSee [26] made a preliminary effort towards eliminating the impact of such background movements. In future, we plan to extend the approach presented in [26] and augment WiID with it to enable user identification from gestures in the presence of background movements.

WiFi Receiver Placement. As discussed in Section 7.6, while WiID achieves acceptable accuracy if the receiver is placed in an adjacent room and that adjacent room does not have significant movements while a user performs predefined gestures, we recommend that to achieve the best performance with WiID, (1) the receiver should be placed in the same room where gesture recognition and user identification is desired, and (2) there should be a line of sight between the user and the receiver. Requiring the WiFi receiver to be in the same room where gesture recognition and user identification is desired is not an unrealistic requirement due to two reasons. First, solutions (such as Google WiFi Mesh [5]) that place multiple access points in different locations inside a house/office are already being used commercially. Second, WiFi receivers have become incredibly cheap, due to which, one can place a simple WiFi receiver in the environment where gesture recognition and user identification is desired at very little cost. The line of sight between the user and receiver is also not a problem when the receiver is placed higher on a wall in the room where gesture recognition and user identification is desired. Nonetheless, we acknowledge that WiID does require LoS for good and stable accuracy and more research is needed to enable user identification through gestures in non line of sight settings where the background movements are pronounced.

Hard Access Control. Recall from Section 1.1 that one of the application scenarios that we mentioned for WiID was soft access control, such as the gestures from a child in a home not being allowed to turn on the TV at the restricted times of the day. While WiID is suitable for such soft access control scenarios, due to the non-zero error rate of WiID, it is not suitable for hard access control scenarios, such as allowing access to a vault in a bank, unlocking a drawer in an office containing sensitive documents, or logging into an administrative account.

9 RELATED WORK

To the best of our knowledge, none of the prior WiFi based gesture recognition systems can identify which user performed the gesture. Next, we give a brief overview of the prior work on wireless signals (including WiFi) based

gesture and activity recognition systems. Note that researchers have proposed several other wireless signal based human sensing systems with objectives such as measuring heart and breathing rates [12], tracking direction of motion of humans behind an obstacle [11], and estimating the number of people in a crowd [38] *etc.* However, as our focus in this paper is on identifying users from their gestures, we first mention the prior work on gesture and activity recognition systems. After that we present the prior work on using WiFi to identify people based on their walking behavior.

We categorize the prior work on wireless signal based gesture and activity recognition systems into two broad categories: commodity device (CD) based and specialized hardware (SH) based. The CD-based systems can be implemented on commodity WiFi devices without requiring any hardware modifications. Our proposed system, WiID, falls under this category. The SH-based systems use software defined radios, such as USRPs [1, 9], often along with specialized hardware, such as directional antennas or custom analog circuits.

CD-based Gesture and Activity Recognition Systems. WiFall detects a user's fall using features such as activity duration and rate of change in CSI values [21]. E-eyes generates histograms of the CSI values and uses them as features to recognize activities such as brushing teeth, showering *etc.* [37]. HeadScan [19] and BodyScan [18] put antennas on user's body. HeadScan recognizes mouth related activities such as coughing and eating using features that include mean, median, and standard deviation in CSI values. BodyScan recognizes activities similar to E-eyes but using the shapes of CDFs of CSI values as features. WiFinger uses DWT coefficients of combined time-series of CSI values to recognize finger gestures that differ in the number of extended and folded fingers [31]. WiGest distinguishes between gestures that give rise to three primitives in RSS values: rising edge, falling edge, and pause [10]. WiDraw tracks the hand of a user by monitoring the changes in the magnitudes of signals arriving at different angles from the hand [30]. WiAG proposed a translation function to enable position and orientation agnostic gesture recognition using WiFi [33]. Unfortunately, none of these systems can identify the user that performed the gesture/activity.

SH-based Gesture and Activity Recognition Systems. AllSee uses an analog envelope-detector to extract the amplitude of the received TV and RFID signals to identify eight gestures performed by a single user at a time [24]. WiSee uses a USRP to extract micro-level doppler shifts in a carrier wave due to human movements to recognize nine different gestures performed by a single user at a time [26]. Just like CD-based systems, these systems can also not identify the user that performed the gesture/activity.

WiFi based Human Identification using Gait. Recently, researchers have proposed several WiFi based systems that identify users by leveraging the observation that different people have different walking gaits. WiWho was the first scheme that proposed to use variations in the CSI measurements to identify humans walking along a straight line at a distance of 1 meter parallel to the line of sight path between the WiFi sender and receiver [39]. WiWho achieved an accuracy of about 80% when distinguishing across 6 users. WiFi-ID [40] appeared at almost the same time as WiWho. WiFi-ID also extracted features from CSI measurements and achieved an accuracy of about 77% across 6 users. WifiU increased the distance of WiWho from 1 meter to 6 meters and the accuracy of WiWho and WiFi-ID from 80% to about 90% for 7 users while using CSI based features [34]. WFID used subcarrier-amplitude frequency features and achieved an average accuracy of a little over 90%, again for individual users walking along straight lines [22]. RAPID used features from both CSI measurements as well as from the sound produced by footsteps to identify users, and reported an accuracy of about 82% for 6 users [16]. Wii extracted both time and frequency domain features from the CSI measurements to identify individual users walking along a straight path and demonstrated an accuracy of a little over 91% [25].

The problem of identifying users from their walk is quite different from the problem that WiID addresses. The objective of WiID is to enable user identification through simple gestures, which can be augmented with WiFi based gesture recognition systems to enable personalized gesture control. The walk based user identification systems do not do that, rather require the user to walk along a certain path before the user can be identifed.

The application scenarios of walk based user identification systems are quite different from the application scenarios of WiID. For example, if the application scenario is to control equipment in one's living room through personalized gestures, WiID is well-suited compared to a walk based user identification system. If the application scenario is to identify people walking in a hallway or to authenticate the person walking up to the entrance of an authorized-personnel-only area, a walk based user identification system is more appropriate compared to WiID. Thus, WiID provides complementary but different functionality from the walk based user identification systems.

10 CONCLUSION AND FUTURE WORK

In this paper, we proposed WiID, a WiFi and gesture based user identification system that can identify the user as soon as he/she performs a predefined gesture at runtime. WiID integrates with the WiFi based gesture recognition systems as an add-on module whose sole objective is to identify the users that perform the predefined gestures. The key technical novelty of WiID lies in leveraging our new observation that the time-series of the frequencies that appear in WiFi channel's frequency response while performing a given gesture are different in the samples of that gesture performed by different users, and are similar in the samples of that gesture performed by the same user. The key technical depth of WiID lies in the algorithm that it uses to extract speed time-series from any given time-series of CSI measurements, and the algorithm that it uses to automatically segment each speed time-series into subseries of appropriate time durations. We extensively evaluated WiID using a large data set of over 25,000 gesture samples and showed that it achieves an average cross-validation accuracy of 92.8% for 5 users across four environments. In future, we plan to study the feasibility of WiID for use in hard authentication scenarios, which will require a strict guarantee that motivated attackers cannot learn and replicate the behaviors of legitimate users. We have not conducted this study as part of this paper because the objective of the work presented in this paper is to enable personalization; hard authentication scenarios may require to completely revisit the problem and redesign the system from the perspective of security.

ACKNOWLEDGMENTS

This work is supported in part by the National Science Foundation, under grant # CNS 1565609.

REFERENCES

[1] [n. d.]. https://www.ettus.com/product/category/USRP-Networked-Series. ([n. d.]).
[2] [n. d.]. Ambient OS. https://www.essential.com/blog/home-now-has-an-os. ([n. d.]).
[3] [n. d.]. Coefficient of Variation. https://en.wikipedia.org/wiki/Coefficient_of_variation. ([n. d.]).
[4] [n. d.]. Contiki: The Open Source OS for the Internet of Things. http://www.contiki-os.org/. ([n. d.]).
[5] [n. d.]. Google WiFi Mesh. https://store.google.com/us/product/google_wifi_learn?hl=en-US. ([n. d.]).
[6] [n. d.]. House Operating System (HOS). http://www.houseoperatingsystem.com/. ([n. d.]).
[7] [n. d.]. Spectrogram. https://en.wikipedia.org/wiki/Spectrogram. ([n. d.]).
[8] [n. d.]. Unity-based normalization. https://en.wikipedia.org/wiki/Normalization_(statistics). ([n. d.]).
[9] [n. d.]. USRP N210. https://www.ettus.com/product/details/UN210-KIT. ([n. d.]).
[10] Heba Abdelnasser, Moustafa Youssef, and Khaled A Harras. 2015. WiGest: A ubiquitous WiFi-based gesture recognition system. In *Proceedings of IEEE INFOCOM*.
[11] Fadel Adib and Dina Katabi. 2013. See Through Walls with WiFi!. In *Proceedings of the ACM SIGCOMM 2013 Conference on SIGCOMM (SIGCOMM '13)*. ACM, New York, NY, USA, 75–86.
[12] Fadel Adib, Hongzi Mao, Zachary Kabelac, Dina Katabi, and Robert C Miller. 2015. Smart homes that monitor breathing and heart rate. In *Proceedings of the 33rd annual ACM conference on human factors in computing systems*. ACM, 837–846.
[13] Kamran Ali, Alex Xiao Liu, Wei Wang, and Muhammad Shahzad. 2015. Keystroke Recognition Using WiFi Signals. In *Proceedings of the 21st Annual International Conference on Mobile Computing and Networking*. ACM, 90–102.
[14] Kamran Ali, Alex X Liu, Wei Wang, and Muhammad Shahzad. 2017. Recognizing Keystrokes Using WiFi Devices. *IEEE Journal on Selected Areas in Communications* 35, 5 (2017), 1175–1190.
[15] Chih-Chung Chang and Chin-Jen Lin. 2011. LIBSVM: a library for support vector machines. *ACM Transactions on Intelligent Systems and Technology* 2, 3 (2011), 27:1–27.

[16] Yuanying Chen, Wei Dong, Yi Gao, Xue Liu, and Tao Gu. 2017. Rapid: a multimodal and device-free approach using noise estimation for robust person identification. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 1, 3 (2017), 41.
[17] Colin Dixon, Ratul Mahajan, Sharad Agarwal, AJ Brush, Bongshin Lee, Stefan Saroiu, and Paramvir Bahl. 2012. An operating system for the home. In *Proceedings of the 9th USENIX conference on Networked Systems Design and Implementation*. USENIX Association, 25–25.
[18] Biyi Fang, Nicholas D Lane, Mi Zhang, Aidan Boran, and Fahim Kawsar. 2016. BodyScan: Enabling radio-based sensing on wearable devices for contactless activity and vital sign monitoring. In *Proceedings of the 14th Annual International Conference on Mobile Systems, Applications, and Services*. ACM, 97–110.
[19] Biyi Fang, Nicholas D Lane, Mi Zhang, and Fahim Kawsar. 2016. Headscan: A wearable system for radio-based sensing of head and mouth-related activities. In *Proceedings of the 15th International Conference on Information Processing in Sensor Networks*. IEEE Press, 21.
[20] Daniel Halperin, Wenjun Hu, Anmol Sheth, and David Wetherall. 2011. Tool Release: Gathering 802.11n Traces with Channel State Information. *ACM SIGCOMM CCR* 41, 1 (Jan. 2011), 53.
[21] Chunmei Han, Kaishun Wu, Yuxi Wang, and Lionel M Ni. 2014. WiFall: Device-free fall detection by wireless networks. In *Proceedings of IEEE INFOCOM*. 271–279.
[22] Feng Hong, Xiang Wang, Yanni Yang, Yuan Zong, Yuliang Zhang, and Zhongwen Guo. 2016. WFID: passive device-free human identification using WiFi signal. In *Proceedings of the 13th International Conference on Mobile and Ubiquitous Systems: Computing, Networking and Services*. ACM, 47–56.
[23] S. Sathiya Keerthi and Chih-Jen Lin. 2003. Asymptotic behaviors of support vector machines with Gaussian kernel. *Neural computation* 15, 7 (2003), 1667–1689.
[24] Bryce Kellogg, Vamsi Talla, and Shyamnath Gollakota. 2014. Bringing Gesture Recognition to All Devices. In *Proceedings of the 11th USENIX Conference on Networked Systems Design and Implementation (NSDI'14)*. USENIX Association, Berkeley, CA, USA, 303–316.
[25] Jiguang Lv, Wu Yang, Dapeng Man, Xiaojiang Du, Miao Yu, and Mohsen Guizani. 2017. Wii: Device-Free Passive Identity Identification via WiFi Signals. In *GLOBECOM 2017-2017 IEEE Global Communications Conference*. IEEE, 1–6.
[26] Qifan Pu, Sidhant Gupta, Shyamnath Gollakota, and Shwetak Patel. 2013. Whole-home Gesture Recognition Using Wireless Signals. In *Proceedings of the 19th ACM MobiCom (MobiCom '13)*. ACM, New York, NY, USA, 27–38.
[27] Bernhard SchÃűlkopf, John C. Platt, John Shawe-Taylor, Alex J. Smola, and Robert C. Williamson. 2001. Estimating the support of a high-dimensional distribution. *Neural computation* 13, 7 (2001), 1443–1471.
[28] Muhammad Shahzad, Alex X Liu, and Arjmand Samuel. 2013. Secure unlocking of mobile touch screen devices by simple gestures: You can see it but you can not do it. In *Proceedings of the 19th ACM MobiCom*. ACM, 39–50.
[29] Muhammad Shahzad, Alex X Liu, and Arjmand Samuel. 2015. Behavior Based Human Authentication on Touch Screen Devices Using Gestures and Signatures. *Mobile Computing, IEEE Transactions on* 23, 1 (2015), 241–254.
[30] Li Sun, Souvik Sen, Dimitrios Koutsonikolas, and Kyu-Han Kim. 2015. Widraw: Enabling hands-free drawing in the air on commodity wifi devices. In *Proceedings of the 21st Annual International Conference on Mobile Computing and Networking*. ACM, 77–89.
[31] Sheng Tan and Jie Yang. 2016. WiFinger: leveraging commodity WiFi for fine-grained finger gesture recognition. In *Proceedings of the 17th ACM International Symposium on Mobile Ad Hoc Networking and Computing*. ACM, 201–210.
[32] Raghav H Venkatnarayan, Griffin Page, and Muhammad Shahzad. 2018. Multi-User Gesture Recognition Using WiFi. In *Proceedings of the 16th Annual International Conference on Mobile Systems, Applications, and Services*. ACM, 401–413.
[33] Aditya Virmani and Muhammad Shahzad. 2017. Position and Orientation Agnostic Gesture Recognition Using WiFi. In *Proceedings of the 15th Annual International Conference on Mobile Systems, Applications, and Services*. ACM, 252–264.
[34] Wei Wang, Alex X Liu, and Muhammad Shahzad. 2016. Gait recognition using wifi signals. In *Proceedings of the 2016 ACM International Joint Conference on Pervasive and Ubiquitous Computing*. ACM, 363–373.
[35] Wei Wang, Alex X Liu, Muhammad Shahzad, Kang Ling, and Sanglu Lu. 2015. Understanding and Modeling of WiFi Signal Based Human Activity Recognition. In *Proceedings of the 21st Annual International Conference on Mobile Computing and Networking*. ACM, 65–76.
[36] Wei Wang, Alex X Liu, Muhammad Shahzad, Kang Ling, and Sanglu Lu. 2017. Device-Free Human Activity Recognition Using Commercial WiFi Devices. *IEEE Journal on Selected Areas in Communications* 35, 5 (2017), 1118–1131.
[37] Yan Wang, Jian Liu, Yingying Chen, Marco Gruteser, Jie Yang, and Hongbo Liu. 2014. E-eyes: In-home Device-free Activity Identification Using Fine-grained WiFi Signatures. In *Proceedings of ACM MobiCom*.
[38] Wei Xi, Jizhong Zhao, Xiang-Yang Li, Kun Zhao, Shaojie Tang, Xue Liu, and Zhiping Jiang. 2014. Electronic Frog Eye: Counting Crowd Using WiFi. In *Proceedings of IEEE INFOCOM*.
[39] Yunze Zeng, Parth H Pathak, and Prasant Mohapatra. 2016. WiWho: wifi-based person identification in smart spaces. In *Proceedings of the 15th International Conference on Information Processing in Sensor Networks*. IEEE Press, 4.
[40] Jin Zhang, Bo Wei, Wen Hu, and Salil S Kanhere. 2016. Wifi-id: Human identification using wifi signal. In *Distributed Computing in Sensor Systems (DCOSS), 2016 International Conference on*. IEEE, 75–82.

Received: February 2018, revised: May 2018, accepted: September 2018