Augmenting User Identification with WiFi based Gesture Recognition

AUTHOR NAMES ANONYMIZED

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 is compatible with all prior (and any future) WiFi based gesture recognition systems and integrates with them 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 21,000 gesture samples. Our results show that WiID achieves an average user identification accuracy of 92.8%.

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

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

ACM Reference Format:
Author names anonymized. 2018. Augmenting User Identification with WiFi based Gesture Recognition. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.* 2, 2, Article 1832 (February 2018), 17 pages. https://doi.org/0000001.0000001

1 INTRODUCTION

1.1 Background

Over the past few years, WiFi based sensing has gained significant popularity and researchers have proposed several WiFi based systems that recognize human gestures [5, 14, 15, 19–21]. 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.

Author's address: Author names anonymized.

2474-9567/2018/02-ART1832 $15.00
https://doi.org/0000001.0000001

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, one user may be interested in controlling the ambience related equipment in an environment while another user may be interested in controlling the multimedia equipment in that same environment using the same set of simple gestures. This can be possible only when, along with recognizing any given gesture, the system can also identify the user that performed it.

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. 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 [21], WiGest [5], or CARM [23], 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. The motivation behind designing WiID in this way is to make it compatible with all prior (and any future) WiFi based gesture recognition systems, and to not tie it to a single gesture recognition system. Consequently, WiID can be integrated with all prior WiFi based gesture recognition systems as an add-on module whose sole objective is to identify the users that perform the predefined gestures.

Figure 1 shows the block diagram demonstrating the general work flow of all prior 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 prior WiFi based gesture recognition systems and then provide an overview of how WiID integrates with any existing WiFi based gesture recognition system. 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.

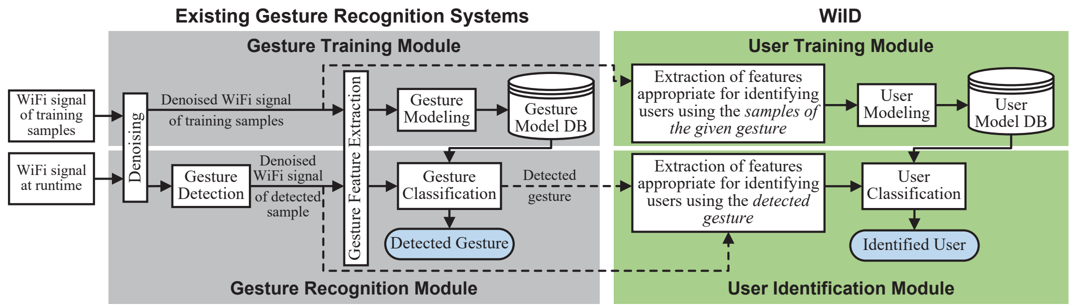


Fig. 1. Block diagram of WiID and how it augments with any exhibiting WiFi based gesture recognition system

Overview of Prior 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 in the block diagram 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 never proposed any modifications to the existing WiFi based gesture recognition systems.

Intuition behind WiID:. The method that WiID employs to identify users is based on the combination of a verified *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 states 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. While we will study the validity of this hypothesis through a WiFi based measurement study in this paper, several existing non-WiFi based gesture recognition systems have already demonstrated that this hypothesis indeed holds true in real world [17, 18]. 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 [23]. 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, out of which we describe two important challenges here and discuss the remaining at appropriate places in the rest of the paper. 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 [3]. 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. In this work, 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. We use coefficient-of-variation [2] to quantify consistency.

1.5 Key Contributions

In this paper, we make following four key contributions.

(1) We proposed, implemented, and evaluated a WiFi based system that can accurately identify the user performing the gestures and is compatible with all prior WiFi based gesture recognition systems.
(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 proposed a method to extract speed time-series from any given time-series of CSI measurements.
(4) We collected a comprehensive data set containing 21,000 samples from 15 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 as an add-on module, uses the PCA based method to remove noise from the CSI streams. The reason for requiring the PCA based denoising is its demonstrated robustness and effectiveness in removing various varieties of noises that are present in CSI streams, as shown by several prior WiFi based sensing systems, such as [21–23]. While most prior gesture recognition systems already use the PCA based method for denoising, a few of them, such as WiGest [5], 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 [23]. As demonstrated in [21, 23], 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 do not provide more details on it in this paper, and refer the interested readers to [21, 23].

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 essentially slides a window over the denoised stream, where at each sliding step of the window, 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 per second, we chose a window size of 1024 samples and a window step size of 5 samples because it gives a suitable frequency resolution of about 1 Hz and a time resolution of 5 ms to trace the changes in the frequencies introduced by the movement of the limb. An FFT at any given window step results in a vector comprising 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 no more than 300Hz [22], 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 set 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 the distortions from the left over noise. A single frequency would appear only in an ideal setting.

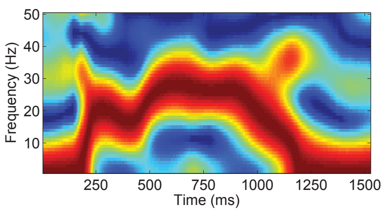


Fig. 2. Spectrogram of a sample

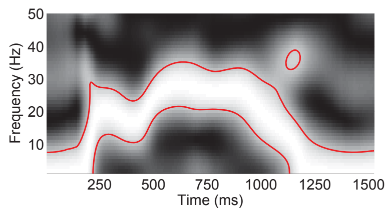


Fig. 3. Contour of high amplitude freqs.

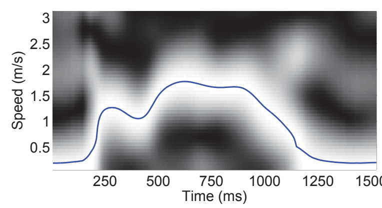


Fig. 4. Speed time-series of Fig. 2

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 90th 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 [22, 23]. The speed time-series obtained from Figures 2 and 3 is shown in Figure 4.

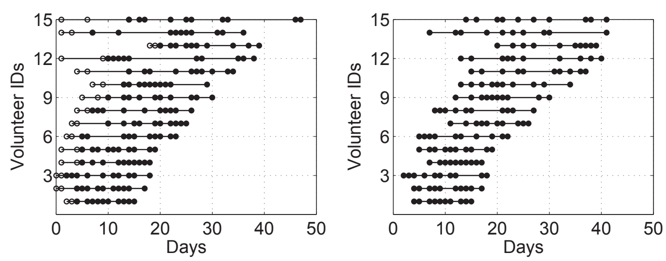
3 DATA COLLECTION AND ANALYSIS

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 set. After that, we use the gesture samples in this data set to study the hypothesis (stated in Section 1.3) 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 21,000 samples from 15 volunteers for 7 gestures in 4 different 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 used the tool presented in [11] 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 achieve a 1000 samples/sec sampling rate. We collected these samples in following four environments (which contained usual furniture): a 400 ft^2 lab, an 96 ft^2 office, a 192 ft^2 living room, and a 154 ft^2 bedroom. We collected these samples after obtaining IRB approval and over a period of 47 days. To incorporate the effects of environmental changes, we randomly moved the furniture in the room each day before collecting samples on that day.

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. As demonstrated in [17, 18], if a volunteer is requested to perform the same gesture a large number of times in a data collection session, the volunteer develops a temporary behavior during that session, which is not his/her true behavior, and is later unable to replicate that behavior. The authors of [17, 18] suggested that to collect any data where user behavior is to be leveraged, do multiple data collection sessions, where each data collection session should be separated by long durations, and in each session, collect only a few samples of each gesture. Based on this suggestion, we asked the volunteers to provide only 5 samples of each gesture in each data collection session, and separated the 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 performing 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 and bedrooms 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 5(a) shows a timing diagram showing the gap between the two try and ten data collection sessions of each volunteer in the lab and office environments. Figure 5(b) shows 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. 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 their. We manually recorded the start and end times of each gesture sample that the volunteers provided.



(a) Lab and office environments (b) Living and bedroom envs.

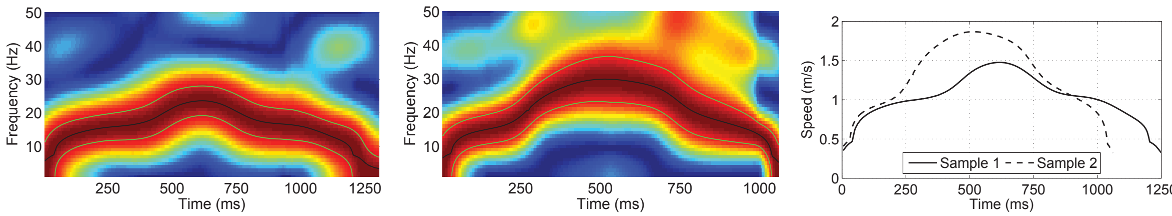
Fig. 5. Timing diagrams of gaps between data collection sessions

3.2 Data Analysis

From our data set, we made following two important observations, which we demonstrate in this section using the samples from our data set.

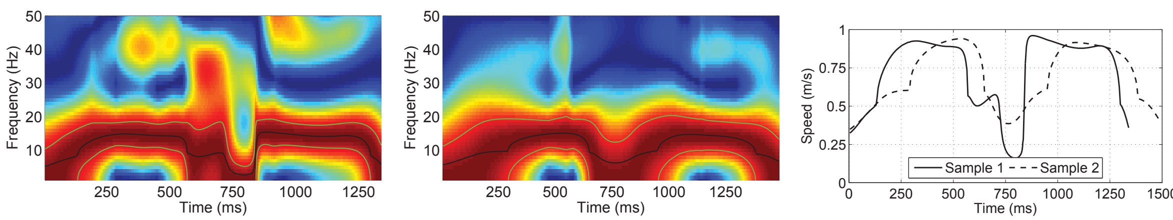
- *Different people perform the same gesture with different behaviors.*
- *An individual user performs the same gesture with same behavior consistently over time.*

Figures 6(a) and 6(b) show the spectrograms of two randomly chosen samples of waving arm (WA) gesture from a volunteer. Figure 6(c) shows the speed time-series obtained from these two spectrograms using the method described in Section 2.3. Figures 7(a), 7(b), and 7(c) show the spectrograms and corresponding speed time-series, respectively, of two randomly chosen samples of the same waving arm (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 very similar observations from the remaining samples in our data set.



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

Fig. 6. Spectrograms and speed time-series of two randomly chosen samples of waving arm (WA) gesture of volunteer 3

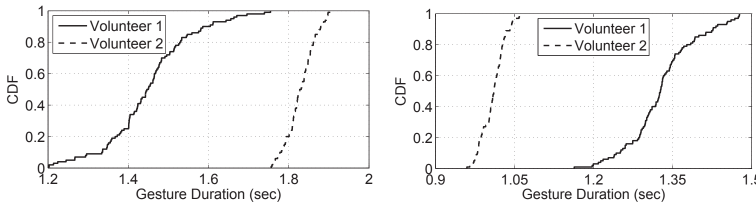


(a) Spectrogram of sample 1 (b) Spectrogram of sample 2 (c) Speed time-series of samples 1 & 2

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

From our data set, we further observed that people take consistent and distinguishing amount of time to perform each gesture. The distributions of gesture durations of different users are centered at different means. Figures 8(a) and 8(b) plot the cumulative distribution functions (CDFs) of the durations of OC and RL gestures, respectively, for two randomly chosen volunteers. These figures show that the overlap in distributions for different users is small and the distributions are centered at different means, which further strengthens our two observations stated at the start of this subsection.

To study how significantly the behaviors of users change over time, we present observations from the samples of the three users who provided us training samples over the longest spans of time among all users. Figure 9 plots the mean and standard deviation of gesture durations (*i.e.*, time to perform a gesture) for WA gesture for each of these three volunteers from the samples in each of the 10 sessions. We observe from this figure that the average gesture duration of each volunteer stays fairly consistent over the sessions, which is an indicator that the user behavior does not change significantly over a span of a few weeks. An interesting observation we make from this figure is that the average gesture duration for one volunteer showed a slightly reducing trend. Even though the decrease in gesture duration is steady, the rate of decrease is very slow. Thus, this slow change in behavior does not create any immediate problems for WiID. To handle such gradually changing behavior, we propose to retrain WiID's classifiers every few weeks. Note that to retrain the classifiers, WiID does not require the users to explicitly provide new training samples, rather it simply uses the samples that it collected from the users at runtime over the past few weeks while the users used the gesture recognition system.



(a) OC Gesture

(b) RL Gesture

Fig. 8. CDFs of gesture durations of two volunteers

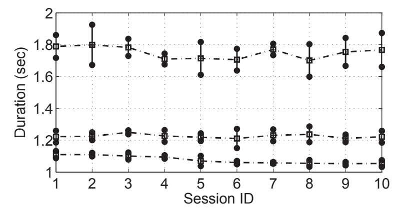


Fig. 9. Average and standard deviation of gesture durations of users over time

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 exact same method to extract and select features from the training samples of any given gesture provided by any given user. Therefore, in the rest of this section, we describe this method considering the samples of an arbitrary gesture i provided by 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 key technical challenges to this goal. The first technical 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 technical challenge is to determine the time duration that is appropriate for use as the segmentation guideline. The third technical 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 for use in classification. However, as different speed time-series usually span different lengths of time, segmenting each time-series into subseries of time 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 a value $\lceil T_k/t_S \rceil$, 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 use the most frequent value, denoted by S, to be the number of subseries that 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 within any given speed time-series have equal time durations. After segmenting all time-series into subseries, we calculate a speed value from each subseries by averaging all speed values within that subseries.

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 10 shows the coefficient-of-variation, 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 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 11 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.

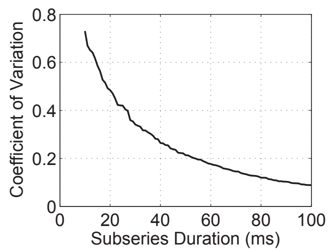


Fig. 10. Impact of subseries duration on cv

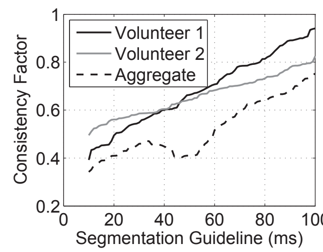


Fig. 11. Impact of t_S on consistency factor

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 11 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 on the speed time-series of each of the 7 gestures collected in each of the 4 environments, and made the same two observations. Therefore, we choose the value of 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 efficiency and 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) [13, 16, 17]. We use the open source implementation of SVDE in libSVM [8].

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, to which WiID is being added to as an add-on module, 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 by 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 of the U classification models is below 50%, then WiID declares that the gesture was not performed by any of the users enrolled with it, rather by someone else.

7 EVALUATION

We implemented WiID on a commodity PC with an 8 core Intel Xeon processor and 16GB RAM. All the evaluations that we present in this section were performed on the data set that we presented in Section 3.1, unless we indicate otherwise. Next, we first present the overall accuracy of WiID. After that, we study the impact of the number of users, the number of gestures, the heights and weights of the users, and the clothing on the accuracy of WiID. Last, we present the processing latency of WiID to study its feasibility for use in realtime.

In all the experiments that we present next, 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, in this section, 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

Recall from Section 3.1 that we collected samples from our volunteers in four different environments. To calculate the overall accuracy of WiID in each of these four environments, 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 average 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 $\times$ 50 samples per gesture = 350) of each volunteer has been tested once. We calculate 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. Consequently, 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 12 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%, 92.7%, 93.4%, and 93.4% in the lab, living room, office, 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. Also recall that when collecting samples, we randomly moved the furniture in the environment each day. Consequently, the CDFs in Figure 12 show the accuracy of WiID while incorporating the changes in user locations as well as static changes in the environment.

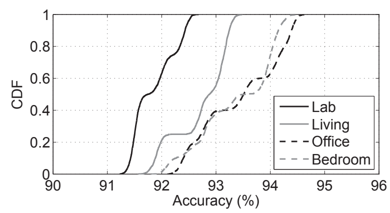


Fig. 12. CDFs of WiID's overall accuracy

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 for eight different values of U, ranging from 3 to 10. Figure 13 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, 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, 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 14 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 14 contains $G + 1$ bars. In each set, the left most

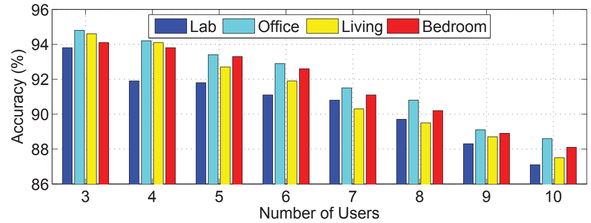


Fig. 13. Impact of the number of users on WiID's accuracy

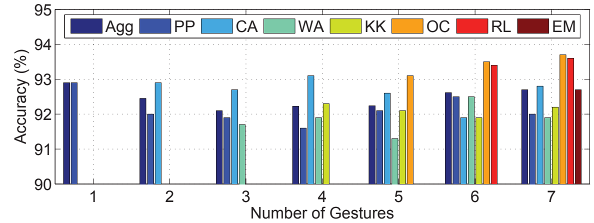


Fig. 14. Impact of number of gestures on WiID's accuracy

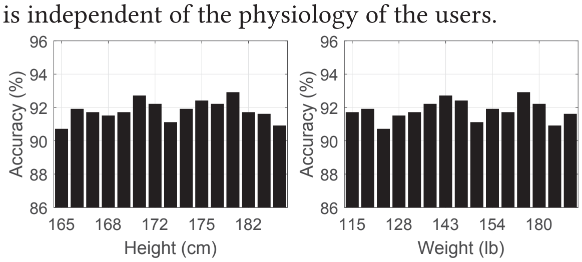
bar represents the average accuracy of WiID in identifying the users across all gestures and all U users, whereas each of the remaining G bars represent the accuracy of WiID in identifying users from one specific gesture. We observe from this figure that the average accuracy of WiID in identifying users 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 15(a) and 15(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, 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.

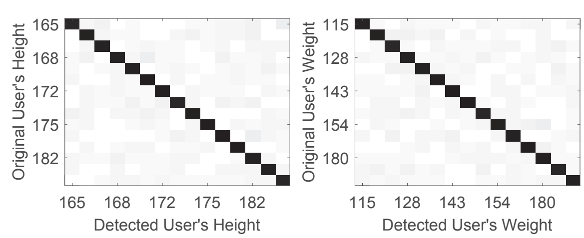
We observe from these two figures that their is no apparent trend in WiID's accuracy with changes in volunteers' heights and weights. This shows that the physiology of the user does not have any significant impact on the accuracy of WiID. We made the same observation from the data collected in the office,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

Figures 16(a) and 16(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. 15. Impact of users' heights and weights on WiID's accuracy



(a) Height (b) Weight

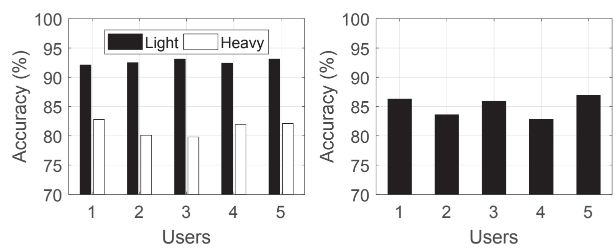
Fig. 16. Confusion matrices of WiID with respect to users' heights and weights

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 17(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. 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 17(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. 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. 17. Impact of clothing on the accuracy of WiID

7.6 Processing Latency

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 seconds. While 13 seconds per user per gesture may seem like a relatively long time, it is practically not problematic because it 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 43 milliseconds 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 $U \times 43$ milliseconds to identify the user from that gesture. Thus, even when $U = 10$, which is already too large a value of U for most real world settings, WiID takes less than half a second to identify a user. This shows that WiID is amenable for use in realtime.

8 RELATED WORK

While most prior WiFi based gesture recognition systems can recognize any predefined gesture at runtime irrespective of the user that performs it, 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 [7], tracking direction of motion of humans behind an obstacle [6], estimating the number of people in a crowd [25], etc. However, as our focus in this paper is on identifying users from their gestures, we only mention the prior work on gesture and activity recognition systems.

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, 4], 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 [12]. E-eyes generates histograms of the CSI values and uses them as features to recognize activities such as brushing teeth, showering *etc.* [24]. HeadScan [10] and BodyScan [9] 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 [20]. WiGest distinguishes between gestures that give rise to three primitives in RSS values: rising edge, falling edge, and pause [5]. WiDraw tracks the hand of a user by monitoring the changes in the magnitudes of signals arriving at different angles from the hand [19]. WiAG proposed a translation function to enable position and orientation agnostic gesture recognition using WiFi [21]. 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 [14]. 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 [15]. Just like CD-based systems, these systems can also not identify the user that performed the gesture/activity.

9 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 is compatible with all prior (and any future) WiFi based gesture recognition systems and integrates with them 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 21,000 gesture samples and showed that it achieves an average user identification accuracy of 92.8%. In future, we plan to study the feasibility of WiID for use in authentication, 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; authentication requires to completely revisit the problem and redesign the system from the perspective of security.

REFERENCES

[1] [n. d.]. https://www.ettus.com/product/category/USRP-Networked-Series. ([n. d.]).

[2] [n. d.]. Coefficient of Variation. https://en.wikipedia.org/wiki/Coefficient_of_variation. ([n. d.]).

[3] [n. d.]. Spectrogram. https://en.wikipedia.org/wiki/Spectrogram. ([n. d.]).

[4] [n. d.]. USRP N210. https://www.ettus.com/product/details/UN210-KIT. ([n. d.]).

[5] Heba Abdelnasser, Moustafa Youssef, and Khaled A Harras. 2015. WiGest: A ubiquitous WiFi-based gesture recognition system. In *Proceedings of IEEE INFOCOM*.

[6] 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. https://doi.org/10.1145/2486001.2486039

[7] 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.

[8] 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.

[9] 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.

[10] 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.

[11] 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.

[12] 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.

[13] S. Sathiya Keerthi and Chih-Jen Lin. 2003. Asymptotic behaviors of support vector machines with Gaussian kernel. *Neural computation* 15, 7 (2003), 1667–1689.

[14] 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. http://dl.acm.org/citation.cfm?id=2616448.2616477

[15] Qifan Pu, Sidhant Gupta, Shyamnath Gollakota, and Shwetak Patel. 2013. Whole-home Gesture Recognition Using Wireless Signals. In *Proceedings of the 19th Annual International Conference on Mobile Computing & Networking (MobiCom '13)*. ACM, New York, NY, USA, 27–38. https://doi.org/10.1145/2500423.2500436

[16] 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.

[17] 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 annual international conference on Mobile computing & networking*. ACM, 39–50.

[18] Muhammad Shahzad, Alex X Liu, and Arjmand Samuel. 2017. Behavior Based Human Authentication on Touch Screen Devices Using Gestures and Signatures. *IEEE Transactions on Mobile Computing* 16, 10 (2017), 2726–2741.

[19] 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.

[20] 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.

[21] 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.

[22] 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.

[23] 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.

[24] 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*.

[25] 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*.