Wi-PT-Hand: Wireless Sensing Based Low-Cost Physical Rehabilitation Tracking for Hand Movements

MD TOUHIDUZZAMAN, STEVEN M. HERNANDEZ, PETER E. PIDCOE, and EYUPHAN BULUT, Virginia Commonwealth University, Richmond, VA, USA

Physical therapy exercises are critically important for the rehabilitation of patients with motor deficits. While these exercises can be most effective when performed properly under the supervision of a physical therapist, it may not be a viable option for all patients. Thus, there is a growing trend towards at-home physical rehabilitation tracking systems as they can be more accessible and flexible for patients. However, existing systems mostly depend on camera and wearable based solutions, which can be costly and limited. To this end, we propose a low-cost and non-intrusive end-to-end solution using IoT-based wireless sensing devices. Our solution, *Wi-PT-Hand*, leverages Channel State Information (CSI) captured from ambient WiFi signals and uses Bayesian optimizers and a hierarchical deep learning model trained to recognize the prescribed hand exercises. The proposed system includes (i) segmentation of the therapy time into activity and non-activity durations, (ii) recognition of the exercise performed in an activity segment, and (iii) counting of the number of repetitions of the exercise performed within that segment. Extensive experimental results show that the proposed system is robust and performs well in various real life scenarios, and thanks to the lightweight design it can work on low-resource edge devices properly.

CCS Concepts: • **Human-centered computing** → **Mobile devices**; **Ubiquitous and mobile computing systems and tools**; *Mobile devices*; • **Applied computing** → **Health informatics**;

Additional Key Words and Phrases: WiFi sensing, physical therapy, rehabilitation tracking, device-free, segmentation

ACM Reference format:
Md Touhiduzzaman, Steven M. Hernandez, Peter E. Pidcoe, and Eyuphan Bulut. 2025. Wi-PT-Hand: Wireless Sensing Based Low-Cost Physical Rehabilitation Tracking for Hand Movements. *ACM Trans. Comput. Healthcare* 6, 1, Article 12 (January 2025), 25 pages.
https://doi.org/10.1145/3688855

1 Introduction

Rehabilitation is the process of recovering a patient's health condition to its normal state after a period of illness. This is a critical process for patients affected by central nervous system disorders such as Parkinson's disease and cerebrovascular diseases (e.g., stroke). For example, stroke affects nearly 800,000 individuals each year in the United States and for approximately 600,000 of them, this is their first event [34]. Many survivors experience persistent difficulty with daily tasks as a direct consequence [27]. These include being less active, being less stable, moving slower as well as adopting inefficient movement patterns resulting in increased physical demands. Thus,

This research was partially supported by the Commonwealth Health Research Board (CHRB) under Award No. 236-12-23.
Authors' Contact Information: Md Touhiduzzaman, Virginia Commonwealth University, Richmond, VA, USA; e-mail: touhiduzzamm@vcu.edu; Steven M. Hernandez, Virginia Commonwealth University, Richmond, VA, USA; e-mail: hernandezsm@vcu.edu; Peter E. Pidcoe, Virginia Commonwealth University, Richmond, VA, USA; e-mail: pepidcoe@vcu.edu; Eyuphan Bulut (corresponding author), Virginia Commonwealth University, Richmond, VA, USA; e-mail: ebulut@vcu.edu.

ACM 2637-8051/2025/1-ART12
https://doi.org/10.1145/3688855

more than two-thirds of stroke survivors receive rehabilitation services after hospitalization. According to recent studies, patients can recover up to 91% functional ability if they start the rehabilitation within 3 months of the stroke [21]. Similarly, rehabilitation can also minimize secondary complications [1]. Thus, maintaining physical activity and performing rehabilitation treatment as early as possible is crucial for the recovery of a patient.

In current practice, rehabilitation treatment for a patient is usually performed under the supervision of a physical therapist who provides guidance to the patient when performing specific exercises and makes sure that each exercise is performed properly for a successful treatment. While such a practice with one-on-one attention from a physical therapist will ensure quality treatment for the patients, it limits the application of rehabilitation treatment to a certain environment and comes with several costs (e.g., treatment cost, trip cost to the facility, dedication of specific time). Thus, alternative solutions which allow patients to perform such rehabilitation activities at home or outside of a dedicated rehabilitation facility can avoid such costs and provide a ubiquitous, and easily accessible solution. On the other hand, these alternative solutions should not reduce the quality of the treatment and must continue to give feedback to patients to ensure that rehabilitation exercises are performed properly and as prescribed. Note that clinical visits and expert based sessions can still be performed as needed, but the count of such visits will get much smaller.

The existing at-home systems mainly depend on wearable sensors or cameras located in the patients' houses [3, 39]. However, such solutions come with several issues. For example, wearable solutions come with several burdens, such as wearing the device, setting it up, and recharging the devices. Moreover, some patients may not be comfortable with wearing them. Similarly, camera-based solutions can trigger privacy concerns as they can record more than what is needed, and they may require certain lighting conditions. Alternative to these existing systems, in this work,[1] we propose an end-to-end at-home WiFi sensing based device-free physical rehabilitation tracking (*Wi-PT-Hand*) system focused on recognizing and measuring hand-based exercises. There are a few other solutions that also rely on the analysis of **radio-frequency (RF)** signals; however, these systems require placements of more costly equipment (e.g., mmWave radar [4]). On the contrary, our goal is to leverage the ubiquitously available WiFi signals and edge devices in most indoor environments and houses for rehabilitation activity tracking, thus providing a low-cost and non-intrusive solution. Moreover, to provide an end-to-end solution, we target a system that can automatically (i) segment the received signal between periods of activity and static periods (i.e., no movement), (ii) classify the specific rehabilitation movement type (e.g., wrist, finger) for each segmented activity, and (iii) count the number of repetitions of the same activity performed within the current activity segment.

The main contributions of this work can be summarized as follows:

- —We design and develop a low-cost, and non-intrusive end-to-end system that can be deployed on edge devices at homes for tracking hand movements.
- —We consider segmentation, classification, and counting components together (for the first time for small scale hand and finger movements) and develop solutions based on Bayesian optimizers and a hierarchical **Dense Neural Network (DNN)** model that are appropriate for edge devices.
- —Through experimental evaluation, we analyze the performance of the proposed complete system as well as its individual components in various scenarios (e.g., different users/environments, hand conditions) and show its practicality and robustness.

The rest of the article is organized as follows. In Section 2, we discuss the related work on both sensor based rehabilitation tracking and WiFi sensing. In Section 3, we present the details of the proposed Wi-PT-Hand system. We evaluate the proposed system through experiments in Section 4. Finally, we provide our concluding remarks in Section 5.

[1]This work extends our preliminary work [19] by focusing on hand-based physical rehabilitation tracking and proposing further techniques such as activity segmentation and counting.

2 Related Work

Our study spans multiple research fields such as rehabilitation tracking, WiFi sensing, and human activity recognition (including segmentation, classification and counting of activities) especially for small-scale activities like wrist and finger movements. Therefore, our analysis of related work is divided into two subsections.

2.1 Sensor-Based Rehabilitation Tracking

Thanks to the ubiquity of Internet of Things devices, there have been many smart health and assistive solutions developed to track the activities of patients during the rehabilitation process. These solutions can be categorized into two major approaches. In the first approach, wearable sensors attached to the body of the patient are considered [7, 8, 28, 29]. These sensors usually contain an Inertial Measurement Unity that can measure accelerometer, gyroscope, and magnetometer readings and can obtain different directional and angular movements of the limbs of the patients. However, patients may feel uncomfortable with wearing these sensors and may not deal with charging them as needed.

In the second approach, the studies use camera-based solutions (e.g., Kinect [42], RGB camera [6], depth camera [22]), and through the analysis of collected frames, detailed patient movements and poses can be detected. While such systems are non-intrusive as they do not require a device worn by the patient, they have certain limitations and drawbacks. For example, they can only track a person when they are in sight of the camera. These systems can also pose a privacy risk to users as they detect not only people's movements but also their faces and the environment they live in. Finally, they can be costly to deploy, especially for large areas where multiple cameras need to be deployed to obtain sufficient coverage.

In addition to these major approaches, there is also a relatively new but growing number of studies that rely on RF signals and radar imaging. These studies [2, 4, 30] rely on specific signals (e.g., FMCW [2], mmWave [4, 30]) and deep learning based analysis of signal features. These solutions can be more effective than previous solutions as even the activities that are performed out of sight can be detected because RF signals can penetrate through some obstacles and reflect from some other objects in the environment. On the other hand, these solutions require special equipment (e.g., mmWave radar), which can be costly. Different from these systems, our solution relies on WiFi signals which can be found in most indoor areas or can be produced through low-cost devices easily. Additionally, the proposed system can use low-cost microcontrollers to capture the signals, preprocess them, and make predictions directly on-board.

2.2 Segmentation and Counting Works on WiFi Sensing

Activity recognition using WiFi sensing has recently been studied a lot with a much focus on detection of different set of activities (e.g., gestures [35], falls [26]) and environmental changes (e.g., soil moisture [17], occupancy counting [24]). We refer the interested readers to the surveys on WiFi sensing (e.g., [23]) and in this part, we specifically cover the WiFi sensing studies that not only aim to recognize the activity but also consider segmentation and counting of activities towards building a complete solution.

We start with summarizing these studies in Table 1 together with their issues and comparison to the proposed work. A device-free fitness assistance system is proposed in [13] for equipment-based exercises. While this study discusses segmentation, and counting components as well, they are not well defined. Auto-correlation based similarity analysis is considered for detecting workout and non-workout times as well as for counting the repetitions in general, but no procedure is defined to find out the exact start and end times of activities. Moreover, no proper evaluation is performed to show the usability of their work. Above all, their system is shown to work only with very large-scale activities like equipment-based exercises. In contrast, we focus on very small-scale hand movements, such as wrist and finger movements, which are more challenging to identify.

The DeepSeg [40] system uses a **Convolutional Neural Network (CNN)** model to identify an activity session's start, motion, end, and static states and thereby segments the **Channel State Information (CSI)** series obtained

Table 1. Comparison of Existing Physical Therapy Monitoring and Tracking Systems

References	Year	Techniques Used	Segm.	Class.	Count.	Issues
Fitness Asst.[13]	2018	Auto-correlation, DNN, Spectrogram analysis	◑	●	◑	Segmentation and counting are not properly described and evaluated.
DeepSeg [40]	2020	CNN to classify states and actions	●	●	○	Stateful segmentation and classification, no counting
RT-Fall [38], Anti-Fall [43]	2016	Band-pass filter to denoise and sliding standard deviation of CSI phase-difference for several states	◑	●	○	Only for fall detection
Temporal Unet [37]	2019	Convolution using Unet to define confidence on standard deviation of CSI amplitudes	◑	●	○	Incomplete description of segmentation and no counting
Tang et al. [32]	2021	Butterworth low-pass filter for segmentation and LSTM-FCN for classification	○	●	○	Only for hand gesture classification
Chen et al. [10]	2021	Butterworth low-pass filter, PCA Dynamic adaptive sliding window for segmentation and CNN for classification	●	●	○	Mostly focuses on whole-body activities
Transtrack [41]	2022	Hampel and band-pass filter, PCA, recursive Otsu thresholding and online meta-transfer learning based on SVM	●	●	○	Finds segmentation and classification for whole-body movements with a single performer
Wi-Gym [44]	2023	Hampel filter, PCA, FFT, Gabor filter, SVM, DTW, Fuzzification	●	●	○	Whole-body actions without counting
U-Shape Nets [36]	2024	DTW and U-Shape deep networks, i.e., FCN, UNET, and UNET++	◑	●	○	Used public datasets of whole body movements of a single person
This study	2024	PCA, Sliding Window, Local Averaging, Bayesian Optimizer, hierarchical DNN, local min and max	●	●	●	N/A

● Fully addressed, ◑ Partially Addressed, ○ Not addressed.

from WiFi sensing over five fine-grained activities like several hand gestures and five coarse-grained activities like running. They divide the entire CSI series into continuous bins of CSI frames and predict which bin is what out of the four states. In this way, the start and end points of a very fine-grained action, like a finger movement, would not be specific and can offset these points by the size of the bin in the worst case. Moreover, the start and end states can be too short to be missed for small-scale activities like finger movement, which reduces the portability of the model for very fine-grained activities like hand physiotherapy actions.

The RT-Fall [38] (and its predecessor, Anti-Fall [43]), also uses stateful transition between activities to segment and detect a fall of various natures. It uses a band-pass filter to denoise the CSI data. Then, the standard deviation of the phase difference is used to identify a person's specific action, which ends with a fall. However, this system is highly dependent on the environment since the phase difference and amplitude of the CSI data vary broadly between environments. In addition, that study focuses on only fall activities.

The Temporal Unet [37] proposes a customized mapping function from CSI series to action label using the convolution of CSI series windows to detect several movements in various environments. The purpose is not entirely segmentation, but their approach produces segments of actions convincingly, which is not fully explored and a proper segmentation methodology is not described. In addition, no repetition counting approach is provided in this study. The action scale is again based on whole-body movements. The study in [32] shows promising classification results with fifty different types of gestures. However, it does not study how to detect the start and end time of gestures, i.e., segmenting, nor how many times the gesture is performed. In [10] a real-time activity

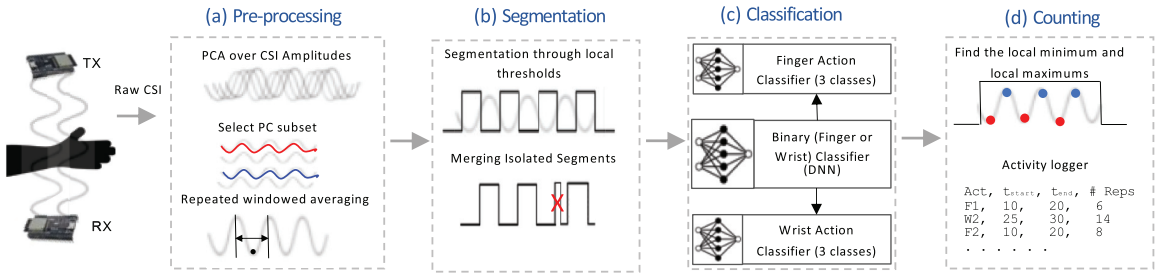


Fig. 1. Wi-PT-Hand system overview.

recognition system is studied using dynamic-sized sliding windows. Their approach looks mostly whole-body movements such as sitting and walking and only one hand-based small scale movement (i.e., waving) is included. Moreover, no solution for counting is proposed.

Similarly, in more recent works [36, 41, 44], some solutions for segmentation and classification of whole-body movements are studied. For example, the study [41] applies Hampel and band-pass filters to denoise **principal components (PCs)** and then takes recursive Otsu thresholding to separate gesture-induced signals from the background noise inspired by image segmentation. In [36], authors compare action recognition, the start and end point detection of action, and action segmentation using three U-shape networks, i.e., FCN, UNet, and UNet++, on several publicly available datasets of whole-body actions. Finally, in Wi-Gym [44], a gymnastics exercise assessment system is developed by comparing how seven gymnastic actions are performed by a person and a trainee through a segmentation and classification procedure. All these studies, however, do not consider small-scale movements and no solution for counting the activities performed is proposed. Overall, to the best of our knowledge, there is no study that considers small-scale hand and finger movements and provides an end-to-end solution with segmentation, classification, and counting components together.

3 Proposed Wi-PT-Hand System

In this section, we introduce our end-to-end wireless sensing based solution for tracking physical therapy hand movements. The proposed system aims to detect (i) start and end of different movement/activity segments, (ii) the type of the movement performed in each segment, and finally (iii) the number of repetitions of that activity performed within that segment. In Figure 1, we provide an overview of the proposed system and illustrate the steps required for reaching out these goals. Next, we elaborate on each of these steps in the following subsections. The notations used throughout the article together with their descriptions are given in Table 2.

3.1 CSI Data Collection

WiFi sensing uses ambient WiFi radio signals to detect and sense the physical properties of the environment. These RF signals propagate over multiple unique physical paths from a transmitter (TX) to a receiver (RX). These multi-paths then cause variations in the RF signal as these signals reflect off of surfaces and propagate through surrounding objects such as furniture, walls and people within the environment.

CSI is a signal metric captured in communication systems that use orthogonal frequency-division multiplexing to allow data symbols to be encoded in multiple subcarrier frequencies allowing for higher symbol throughput and resilience to signal fading and shadowing caused by multi-path interference in the channel. CSI is modeled using $y^{(i)} = H^{(i)}x^{(i)} + \eta^{(i)}$, where i is the subcarrier index, x is the transmitted signal, y is the received signal, η is a noise vector, and H is a complex vector containing the CSI denoting the transformation change required from the input x to the output y. The complex CSI vector (in 802.11) contains 64 subcarriers where 52 are data subcarriers and 12 are null subcarriers. The CSI value for each subcarrier is defined as a complex number with a real ($H_r^{(i)}$)

Table 2. Notations and Their Descriptions

Notation	Description
H	CSI series
ρ_s	First selected PC from the beginning
ρ	Number of total selected PCs
η	Number of repetitive averages
ω	Window size
τ	Threshold used for the separation of activity and non-activity segments
σ	True labels per CSI frame
P_{pca}	PCA output for CSI amplitudes
P	Average overall selected PCs of P_{pca}
P_ω	Preprocessed CSI data over a window size ω
ψ	Window multiplier
γ	Smoothing factor
$\mathcal{A}$	Action label for a CSI frame or segment
$\mathcal{A}'$	Non-action label for a CSI frame or segment
χ	Array of activity and non-activity labels per CSI frame
κ	Set of counts per activity segment
$\lvert X\rvert$	Size or length of any array X
$\overline{X[a:b]}$	Mean of values within range $[a, b]$ in any array X

and an imaginary ($H_{im}^{(i)}$) component, which can be used to compute amplitude, $A^{(i)} = \sqrt{(H_{im}^{(i)})^2 + (H_r^{(i)})^2}$, and phase, $\phi^{(i)} = atan2(H_{im}^{(i)}, H_r^{(i)})$.

To collect CSI data, we use the WiFi-enabled ESP32 microcontrollers and our recently developed open-source ESP32-CSI Toolkit [15]. These ESP32 microcontrollers provide a small-size, low-cost, and standalone solution compared to other existing methods [5, 14, 25] (which require a host laptop with an updated Network Interface Card); thus, they can be easily deployed anywhere. We transmit frames from an ESP32 transmitter and then collect WiFi CSI from a separate ESP32 receiver at a fixed rate of 100Hz. We then extract the amplitudes of the CSI data to be used in the next steps of the proposed system.

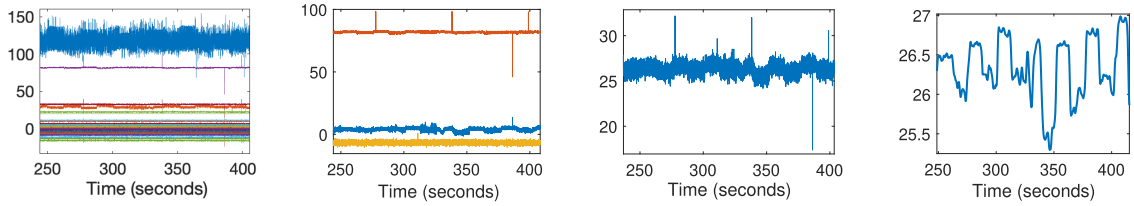
3.2 Data Preprocessing

Once the CSI data is collected, we then preprocess it through the steps given in Algorithm 1. That is, we first run **Principal Component Analysis (PCA)** over the amplitudes of the CSI data (line 1). Then, from the set of PCs computed by PCA, we select a smaller subset of ρ components starting from the ρ_s^{th} component (line 2). These components are considered to be the most significant representative consecutive subset for the data and will be determined by the optimizer. Later on, we take the average of the selected PCs to further reduce the dimension of the data and make the impact much more visible (line 2). Finally, we slide a window over the entire time series data and perform a moving average over this data for η times recursively using a window size of ω. As discussed in Section 3.6, the values of η and ω are optimally chosen along with a few other parameters using a hyperparameter optimization approach in our system.

Figure 2 shows how Algorithm 1 preprocesses the series of PCs of the CSI time series data step by step. The output of PCA over the amplitudes of the CSI data consists of 64 PCs as shown in Figure 2(a). For this specific dataset, our hyperparameter optimization method chooses the third, fourth, and fifth PCs as the best subset from all PCs, as depicted in Figure 2(b). By taking the average of amplitudes in these selected PCs, the impact gets more

Algorithm 1: Preprocessing of Data (H, ρ_s, ρ, η, ω)

$P_{pca} \leftarrow PCA(H)$
$P \leftarrow \overline{P_{pca}[\rho_s : \rho_s + \rho - 1]}$ `// Select a subset of PCs and take the average`
for $n \leftarrow 1$ **to** η **do**
  for $i \leftarrow 1$ **to** $|H|$ **by** 1 **do**
    `// Select a sliding window`
    $a \leftarrow \max(1, i - \omega/2)$
    $b \leftarrow \min(|H|, i + \omega/2)$
    $P_\omega[i] \leftarrow \overline{P[a : b]}$ `// Moving average`
  $P \leftarrow P_\omega$
return P_ω



(a) All PCs (P_{pca}) (b) Selected $\rho = 3$ PCs (c) Avg. of the selected PCs (P) (d) Repetitive average (P_ω)

Fig. 2. Step by step impacts of Algorithm 1.

visible while having some noise, as shown in Figure 2(c). Furthermore, with repetitive and recursive averaging for η times over sliding windows of size ω, the noise is filtered as plotted in Figure 2(d). After this preprocessing part, we prepare to segment and label this result for the activity and non-activity portions.

3.3 Activity Segmentation

The trending nature of fluctuation of P_ω obtained after the preprocessing steps can be seen very much related to the actions performed. Thus, we have designed our segmentation method based on this observation. To this end, we consider two different approaches. In the first approach, we pick the **overall average (OA)**, τ, of the series P_ω, as a threshold to separate the activity and non-activity segments using the following equation:

$$\forall i \in [1, |P_\omega|], \chi_i = \begin{cases} \mathcal{A}, & \text{if } P_\omega[i] < \tau \\ \mathcal{A}', & \text{otherwise} \end{cases}, \text{where } \tau = \overline{P_\omega}. \tag{1}$$

Here, $\chi_i = \mathcal{A}$ and $\chi_i = \mathcal{A}'$ represent an activity and non-activity CSI frame, respectively. The average value, $\tau = \overline{P_\omega}$ is considered as the threshold over the series P_ω, which is basically a flat line along the mean of the data series. It identifies the portion of the data above this threshold as non-activity and the portion of the data below the threshold as activity, which aligns with our observation. While this OA value can provide some understanding toward the separation of activity and non-activity periods, due to many factors (e.g., environment, hardware), there usually occurs local variations in the data; thus OA method often fails to identify such local trends of the time series data with such a global average value. This can be observed in Figure 3, which clearly shows that the globally calculated OA cannot be used for identifying all activity segments while a **local average (LA)** might be more successful to differentiate activity and non-activity periods. To this end, as a second approach, we propose to

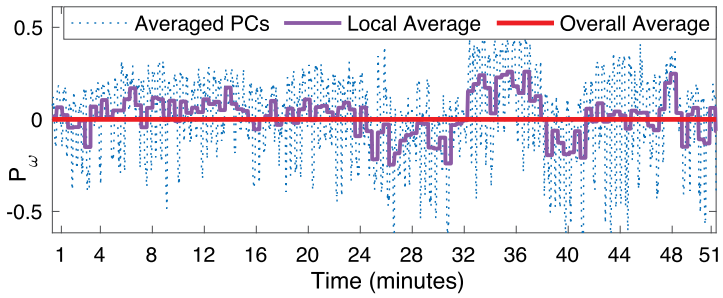
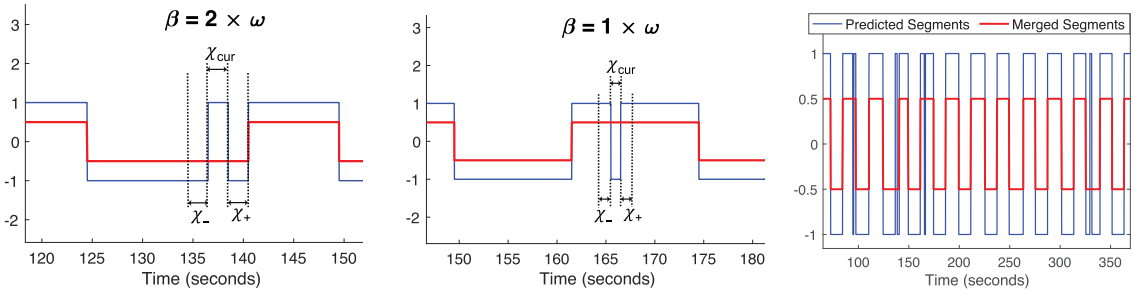


Fig. 3. Sinusoidal trending nature of the PCs of the CSI data, motivating choice of LA as the threshold rather than OA as the threshold.



(a) Merging divided non-activity segment (b) Merging divided activity segment (c) Fixing Multiple Isolated Segments

Fig. 4. Examples depicting Algorithm 3 in action.

use a threshold obtained from a LA over a larger window of size $\psi \times \omega$, and we aim to recognize local changes in the series by comparing this LA in a small and a large window. The details of this approach is given in Algorithm 2. We first calculate the thresholds in large non-overlapping windows (lines 2–4). Then, if the LA obtained in the smaller window (that the current data point is in) is less than the LA in the larger window (line 7), we recognize it as part of an activity segment; otherwise, it is considered within a non-activity segment.

The second approach is more capable of recognizing the activity segments throughout a long data series compared to the first OA based approach. However, both of these methods can produce isolated segments with small durations, usually on the edge of the actual segments. This can be within an activity segment (as shown in Figure 4(a)) or within a non-activity segment (as shown in Figure 4(b)). To address this problem, we develop a merging process for such segments as described in Algorithm 3. That is, starting from the smallest possible segment size ω to the maximum possible segment size $\omega \times \gamma$, we check all possible segments of size β over the entire time frame. If each of the current, preceding and succeeding windows of the same size can be uniformly considered as either an activity ($\mathcal{A}$) or non-activity ($\mathcal{A}'$) segment, then the current segment assignment is considered to be valid. But if the current segment does not match with the preceding and succeeding segments (line 6), which also requires them to be the same segment type (i.e., $\mathcal{A}$ or $\mathcal{A}'$), the current segment is converted to its neighbors' type (line 7). Note that the rationale behind starting with smaller possible segment sizes (line 1) is this bottom-up approach allows merging smaller falsely identified segments first, leading it to generate larger mergeable segments.

Figure 4 illustrates examples of this merging process. In Figure 4(a), a misidentified activity segment is removed as it is surrounded by non-activity segments of same size, while in Figure 4(b) a misidentified non-activity segment

Algorithm 2: Segmentation of Activities (P_ω, ω, ψ)

$\omega_m \leftarrow \psi \times \omega$ // larger window
// Find local thresholds by averaging in non-overlapping large windows
for $i \leftarrow 1$ **to** $|P_\omega|$ **by** ω_m **do**
  $s \leftarrow max(1, i - \omega_m/2)$, $e \leftarrow min(|P_\omega|, i + \omega_m/2)$
  $\tau_s[s:e] \leftarrow \overline{P_\omega[s:e]}$
// Assign activity labels by comparing smaller window and larger window averages
for $i \leftarrow 1$ **to** $|P_\omega|$ **by** ω **do**
  $s \leftarrow max(1, i - \omega/2)$, $e \leftarrow min(|P_\omega|, i + \omega/2)$
  if $\overline{P_\omega[s:e]} < \tau_s(\frac{s+e}{2})$ **then**
    $\chi[s:e] \leftarrow \mathcal{A}$ // Activity
  else
    $\chi[s:e] \leftarrow \mathcal{A}'$ // No activity
return χ // Activity/non-activity labels

Algorithm 3: Merging Isolated Segments (χ, ω, γ)

for $\beta \leftarrow \omega$ **to** $\gamma \times \omega$ **by** ω **do**
  for $s \leftarrow (\beta + 1)$ **to** $(|\chi| - 2\beta + 1)$ **by** 1 **do**
    $\chi_{cur} \leftarrow \overline{\chi[s:s+\beta-1]}$
    $\chi_- \leftarrow \overline{\chi[s-\beta:s-1]}$ // preceding segment
    $\chi_+ \leftarrow \overline{\chi[s+\beta:s+2\beta-1]}$ // succeeding segment
    // If the current segment does not match with neighbors, it is converted.
    if $\big((\chi_{cur} = \mathcal{A} \,||\, \chi_{cur} = \mathcal{A}') \,\&\, (\chi_- = \mathcal{A} || \, \chi_- = \mathcal{A}') \,\&\, (\chi_+ = \mathcal{A} || \, \chi_+ = \mathcal{A}') \,\&\, (\chi_{cur} \neq \chi_- \& \, \chi_{cur} \neq \chi_+)\big)$ **then**
      $\chi[s:e] \leftarrow \chi_+$
return χ

(which divides the activity segment) is converted to an activity segment as it is surrounded by activity segments. Finally, in Figure 4(c), the result of running the merging algorithm on multiple segments over a time frame can be seen. All the isolated and falsely identified segments are merged or removed properly. Note that the parameters used both in Algorithms 2 and 3 are optimized through an optimizer as we will discuss later. The segmentation accuracy is defined as the number of activity or non-activity CSI frames that are correctly identified, and the optimizer uses the corresponding F1-loss for this.

3.4 Activity Classification

After the segmentation of the CSI time series into activity and non-activity segments is completed, next, we start the activity classification process. That is, for each activity segment, we need to recognize which specific activity is performed. Here, considering the two different subcategories of the hand-based movements, namely, wrist-based and finger-based movements, we propose a hierarchical classification model. That is, we first aim to recognize the subgroup of the action performed, and then we aim to recognize the actual activity type within that subgroup. To this end, for the first step, we develop a binary classification model, and for the second step, we develop two different intra-group activity recognition models. The goal of this hierarchical approach is to develop

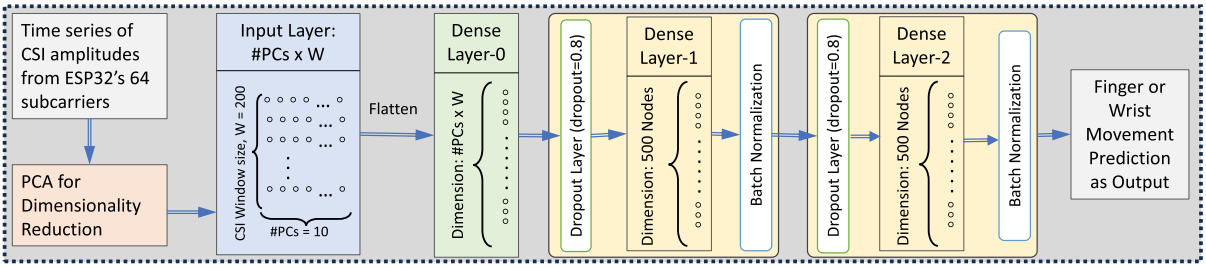


Fig. 5. Activity group binary (finger or wrist) classifier DNN model specifications out of the used three DNN models.

a more scalable solution so that new categories of activities can easily be added later without having to retrain the entire model again each time.

For each of these models, we design a machine learning classifier model $\mathcal{M}$ using a DNN architecture with four dense layers, and we set the number of output neurons equal to the number of classes to be recognized (e.g., two classes for the binary model that aims to recognize the subcategory). We apply a dropout layer between each dense layer to prevent overfitting. Finally, we use the Adam optimizer to optimize the categorical cross entropy loss function,

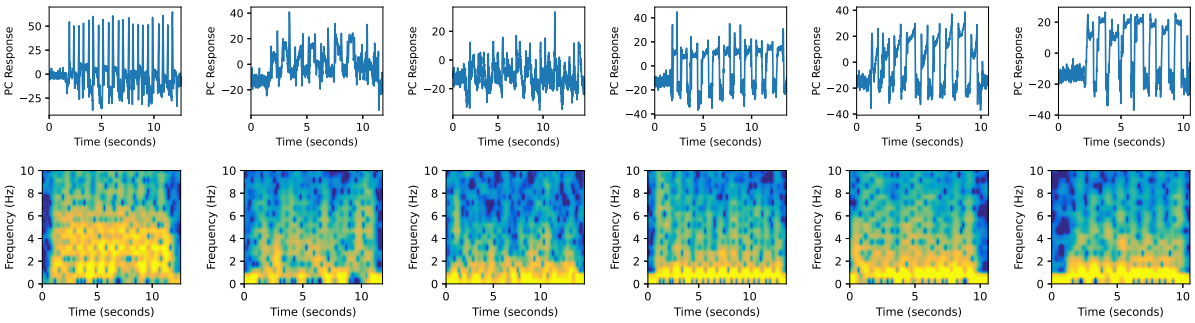
$$\mathcal{L}(x, y) = \frac{1}{N} \sum_{i=1}^{N} \sum_{c=1}^{C} y_i[c] \log \left(\mathcal{M}(x_i)[c]\right),$$

where $\mathcal{M}(x_i)$ is the model prediction for input CSI x_i and y_i is the one-hot encoded true class for the ith CSI measurement. Training is performed on batches of size 1,024 leveraging only the first 10 PCs which are precomputed using PCA. An optimized model for the initial binary classification model is depicted in Figure 5 along with the evaluated hyperparameters. The other two models for specific and intra-group wrist and finger action classification are similar to this model, except for different values of the hyperparameters. Note that more complicated models can be considered for classification tasks in this case, however, our motivation is to keep the models as lightweight as possible to fit them into edge devices.

3.5 Activity Repetition Counting

Once the CSI data series is segmented into activity segments and each activity segment is classified as one of the finger or wrist activities, the next step is to count how many times the same activity is repeated within each activity segment. To this end, we use the corresponding activity segment's internal periodic behavior and similarity.

In Figure 6, we illustrate this periodic behavior through the line plots and spectrograms for the first PC's response for six different activities that we also use for our evaluations. Each subfigure clearly shows that there is a repetitive signal pattern which is also confirmed to closely match with the number of repetitions of the activity. Motivated by this relation, we then use the total number of local optima, i.e., local maxima and local minima, within the segmented region to count the number of repetitions of the activity. Here, note that the raw CSI data, even after averaging over a selected number of PCs (shown as Avg. of PCs, P, in Figure 7) can include so many local optima points looking like noise. The repetitive averaging (η times) method over an optimal window size (ω) can help obtain the signal pattern (i.e., P_ω) that matches with the actual counts. However, this averaging should be optimized for the counting specifically (i.e., preprocessing steps used for segmentation should be performed again for counting). Moreover, as it is shown in Figure 6, the repeating patterns of different activities can be different thus, the optimization model for getting the activity counts should be uniquely developed for each



(a) Pitch Action (b) Roll Action (c) Yaw Action (d) Fist (e) OK Sign (f) V Sign

Fig. 6. Line plots and spectrograms for the first PC response showing unique repetitive patterns of different hand and finger movements.

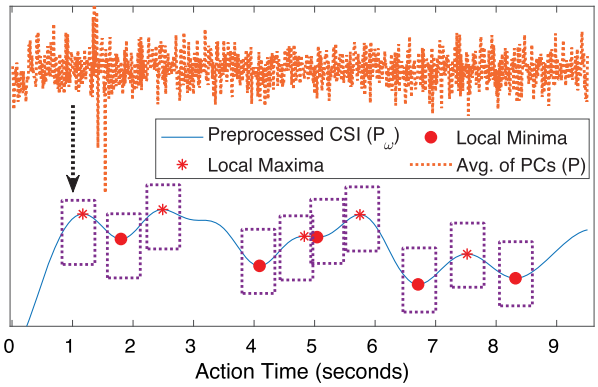


Fig. 7. Counting 10 repetitions of an action by identifying the total local optima within a specified activity segment after an optimized number of repetitive window averaging.

activity. Bringing all these together, we define the counting process (i.e., finding the total number of local optima in P_ω) after an optimized preprocessing (Algorithm 1) is performed for each of the experimented actions.

For counting error, which is used during optimization of the model, we use the **mean absolute error (MAE)**, which is defined as:

$$\mathrm{E}_c = \frac{1}{n}\left(\sum_{i=1}^{n} |\mathrm{Y}_i - \hat{\mathrm{Y}}_i|\right), \tag{2}$$

where Y_i and $\hat{\mathrm{Y}}_i$ are the actual and predicted number of repetitions of the same activity in the input CSI segment i, respectively.

3.6 Hyperparameter Optimizer

Next, we elaborate on the process used to optimize the values of the hyperparameters used at different parts of the proposed system. The optimization process is illustrated through a flow chart shown in Figure 8, where each of the segmentation, classification and counting processes is handled separately, but one's output is used in the other one as input.

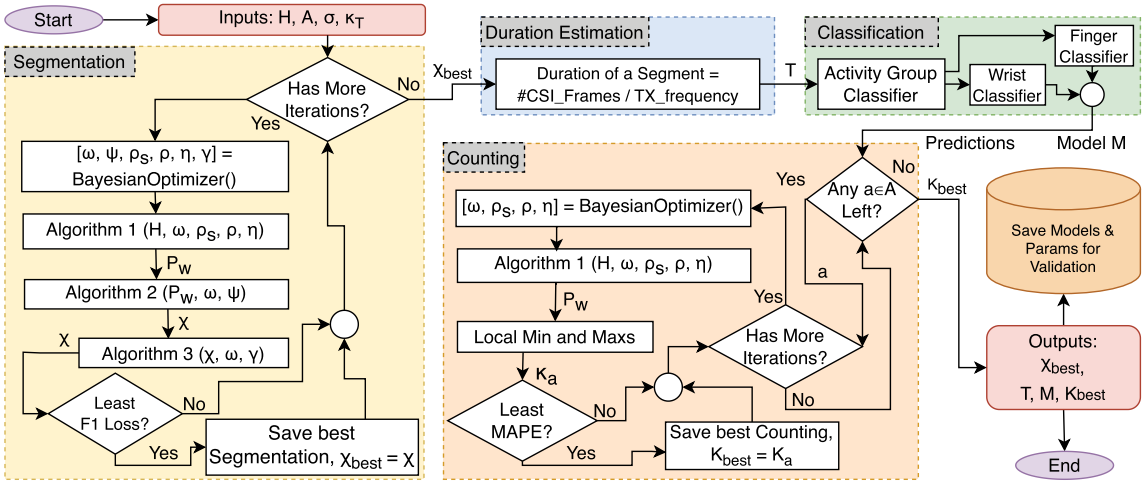


Fig. 8. Flowchart of the full system with optimization iterations.

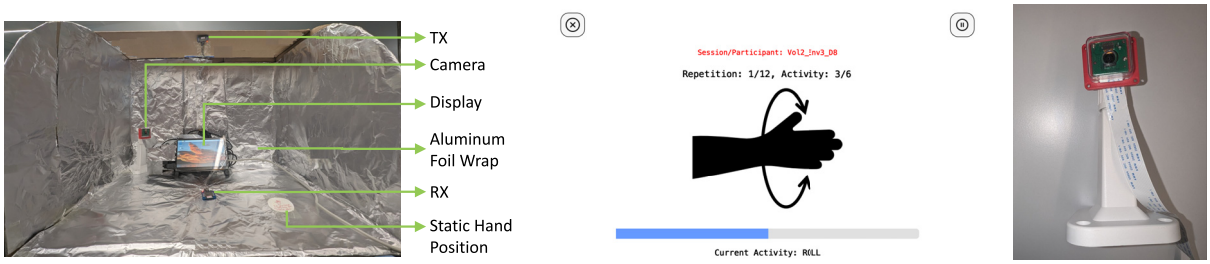
To begin with, we get the CSI series (H), set of the actions ($\mathcal{A}$), true labels per CSI frame (σ), and true counts per action segment (κ). For segmentation part, we use Algorithms 1, 2, and 3 in sequence and predict the segmentation value, χ, which is basically activity or non-activity label for each of the CSI frame out of the provided CSI series, H. The values of parameters (e.g., window size (ω), window multiplier (ψ), group of PCs (defined by ρ_s and ρ), number of repetitive averages (η)) used in these three algorithms are optimized through a Bayesian optimizer, which uses the true segmentation labels per CSI frame, σ, and the F1-loss of each prediction made to reach the best set of parameter values through several iterations (e.g., 1,000 as used in our evaluations). Note that as our system has a specific and stable transmission (TX) frequency, i.e., 100 Hz, we can also calculate the duration of any activity or non-activity segment by dividing the total number of CSI frames within that specific segment by the TX frequency.

For the DNN models used for classification as described in Section 3.4, we leverage **Tree-structured Parzen Estimator (TPE)** as elaborated in [16] to optimize the hyperparameters. These parameters include CSI window size (50–1,000 CSI frames per window) passed to the model, number of hidden neurons ([25–500]), learning rate [10^{-9}–10], whether or not to apply kernel or activation regularization and the per layer dropout percentage [0.0–0.9]. Note that for the proposed hierarchical model, we can train and perform hyperparameter optimization for each sub-model independently of one another. Thus, this allows for greater control of the training process of each sub-model and offers the ability to add or replace sub-models without affecting the prediction quality of the rest of the hierarchical system.

Finally, in the counting part, we use a Bayesian optimizer as in the segmentation section, for the set of input parameters (i.e., ω, ρ_s, ρ, and η) required toward obtaining the best counting accuracy. Note that this is performed for each activity separately, as the periodic behaviors of different actions can be different due to the intrinsic unique features of the activity. Each of these optimizations is run for 1,000 iterations and the sets with the least MAE, as described in Equation (2), are saved as κ_{best}. At the end of this optimization process, the best sets of hyperparameters as optimized by the Bayesian optimizer (for segmentation and counting) and the TPE (for DNN), are saved to provide validations with different datasets afterward.

4 Evaluation

In this section, we start by describing the experimental setup and the datasets collected. We then evaluate the performance of the proposed system through results regarding different individual components, namely,



(a) (b) (c)

Fig. 9. Data collection box with (a) the locations of all components, (b) a display to show animated instructions/predictions, and (c) a camera to record the ground truths of segmentation, counting, and classification.

(a) (b)

Fig. 10. Experimental design with TX/RX pairs illustrated for two different categories of activities (a) hand and wrist movements and (b) finger movements.

segmentation, activity classification, and counting. We then evaluate the system as a whole and compare it to an existing system.

4.1 Experimental Setup

To evaluate the proposed *Wi-PT-Hand* system, we performed our experiments using a pair of ESP32 microcontroller units. These devices are low-cost and can be used to collect CSI data through the tool we developed recently [15]. One ESP32 is used in the TX role and the other in the RX role. The sampling rate is set as 100 Hz. They are aligned vertically in a box (with one side open), where the user's hand performs the activities. The distance between the TX and RX devices was 35 cm. The hand is placed in the line of sight of the TX-RX pair. Figure 9 shows the equipment setup. As shown in Figure 9(a), we wrapped the three surrounding walls and the bottom of the box with aluminum foil to mitigate external electromagnetic interference from the environment and to minimize the impact on the optimized algorithms and machine learning models. A Raspberry Pi supported display is placed inside the box to show instructions to the volunteer to perform specific actions during the data collection session using our annotator app [33] in collaboration with our server app [18] to save the data from ESP32 inside the same Raspberry Pi device. This display can also be used to show predictions in a practical deployment. Besides, a camera is placed to record the movements inside the box so that we can prepare the ground truths of segmentation, counting, and classification. Figure 10 shows the movements performed inside our setup.

4.2 Hand Movement Datasets

We have collected CSI measurements for two types of physical therapy procedures (i) wrist movements and (ii) finger movements. Throughout the evaluation section, we will mainly present the detailed results obtained for the first volunteer's data only. However, to show the generalizability of our system, we test the setup in five different environments as shown in Table 2(b) and, as per Table 2(c), with ten volunteers of age groups ranging from 19

Table 3. Information About the Datasets

(a) Activity durations and repetitions in training ($D1$), validation ($D2$) and test (other volunteers) datasets

Dataset	Reps.	Action Dur.	Session Dur.
D1	20	5–15 seconds	41.4 minutes
D2	4	5–15 seconds	8.1 minutes
Others	12	5–15 seconds	30 minutes

(b) Environments

No.	Location
1	Conference room
2	Open lab area
3	Living room
4	Bedroom
5	Office room

(c) Categorical Variations of the Volunteers

Category	Variations
Age group	19–67 years
Male	7 volunteers
Female	3 volunteers
Conditions	3 types

to 67 years, of both male and female genders and with three types of hand conditions, i.e., normal condition, wearing gloves, and wearing a wrist and thumb support given by physiotherapist to real patients. Overall, we collected 20 different datasets.[2]

4.2.1 Wrist Movements. The wrist movements are pitch (P), yaw (Y), and roll (R) of the palm around the wrist, as illustrated in Figure 10(a). These three movements are typically used to test the movement capability of the wrist along three axes during physical therapy, respectively. The volunteer repeated each hand movement multiple times for a specific duration measured in seconds. There were complete stops in between each successive action.

4.2.2 Finger Movements. We consider small-scale finger movements as the next set of actions. These finger movements are OK sign (O), Victory sign (V), and Fist (F) as illustrated in Figure 10(b). These finger movements are specifically selected to cover the exercise of almost all the joints in the fingers of a human hand.

The collected data is divided into two, to be used in training and validation of our algorithms using these hand movements. Table 3 provides an overview of these two sub datasets. In order to comply with a practical scenario, the actions were performed in random order, and the duration of each action was varied within a specific range of 5–15 seconds. The training dataset ($D1$) in this environment has 20 sessions of each of the six actions which includes several repetitions of that action. The validation dataset ($D2$) has a similar setup but it includes 4 sessions of each action. The training dataset has a total duration of 41 minutes and 25 seconds, and the validation dataset has an entire duration of 8 minutes and 6 seconds. The datasets collected in different environments and/or with different volunteers include 12 repetitions (each with 5–15 seconds) of each hand movement with a total duration of 30 minutes.

4.3 Results

As Wi-PT-Hand system has different components (i.e., segmentation, classification, counting), as shown in Figure 1, we evaluate the performance of each component separately, as well as the entire system's performance. For the Bayesian optimizers used in segmentation and counting parts, we use the implementation available in Matlab 2022a version [9, 12, 31]. We also use Keras library [11] to implement the DNN models used in the classification part.

4.3.1 Activity Segmentation. Segmentation results are generated for both OA and LA methods in three different ways. In the first way, we use the dataset D1 for both training (of the optimizer) and validation; in the second way, we use the dataset D1 for training and D2 for validation; and finally, we use the dataset D2 for both training and validation. We consider these different ways to observe how the results change for our method in different scenarios and how generalizable our results are even within one volunteer's data.

Table 4 shows the segmentation results for all these different ways and different methods. Overall, LA method in general can provide better segmentation accuracy than OA method, thanks to its varying and sliding window

[2]These datasets and our codebase can be found in https://github.com/MoWiNG-Lab/wipt

Table 4. Segmentation Results Through Different Methods and Training/Validation Datasets

Method	Train/ Validate	ω	ψ	ρ_s	ρ	η	γ	Accuracy (%) by CSI-Frames	F1-Score	Segmentation Number			
										Accuracy(%)	FP	FN	Total
OA	D1/D1	50	-	2	2	3	3	81.92%	80.57%	90.45%	14	9	241
OA	D1/D2							84.65%	82.84%	83.67%	4	4	49
OA	D2/D2	50	-	2	2	3	4	92.07%	91.03%	100%	0	0	49
LA	D1/D1	100	19	2	2	5	2	91.41%	90.37%	97.51%	4	2	241
LA	D1/D2							90.27%	88.94%	100%	0	0	49
LA	D2/D2	50	27	2	27	10	7	94.89%	94.42%	100%	0	0	49

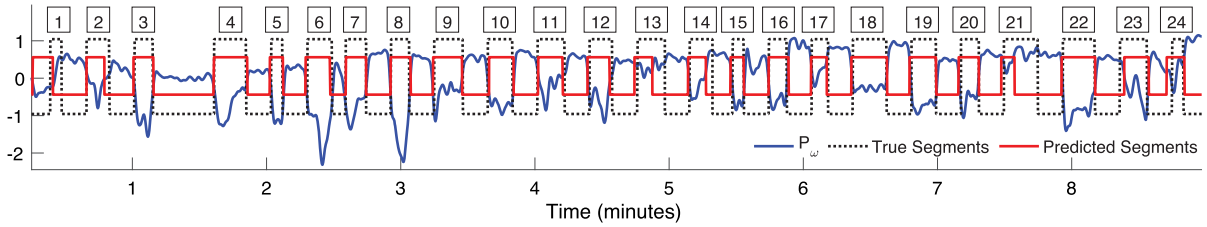


Fig. 11. Segmentation results with LA method optimized/trained with dataset D1 and validated with dataset D2 (i.e., D1/D2) yielding 90.27% per CSI frame segmentation accuracy and 100% segment detection accuracy.

based approach that takes into account local changes in the CSI data. In addition to the segmentation accuracy, we also consider accuracy in terms of the percentage of the number of whole segments (including both the activity and non-activity segments) correctly identified compared to the total number of segments (shown under the "Total" column) in the validation dataset. We consider the non-action segments being wrongly predicted as action segments to be **false-positive (FP)** segments, while the action segments being wrongly predicted as non-action segments are considered **false-negative (FN)** segments. If the number of total segments is S, the number of FP segments is S_{FP} and the number of FN segments is S_{FN}, then this accuracy is given as, $A_s = (1 - (S_{FP} + S_{FN})/S) \times 100\%$. The results in Table 4 show that LA method provides almost perfect accuracy in all scenarios and it is much better than what is obtained when OA method is used. Since the D1/D2 case is the most realistic scenario, we observe a clear difference there (100% vs. 83.67%).

In Table 4, we also show the optimized values of each of the parameters. The optimizer finds the window size of 50 or 100 CSI frames as the most effective window size. The second and third PCs are found to be the most significant ones yielding the best results regarding segmentation. The number of repetitive averages is generally within the range of 3–5, while the merging factor performs better with lower values. It is worth mentioning that the most time-consuming part of our system is Algorithm 3. Therefore, the lower the merging factor remains, the faster the system yields results.

In Figure 11, we show the segmentation results specifically in D1/D2 case to visualize how the segments obtained by the proposed LA method align with the actual segments. There are a total of 24 activity segments (and 25 static segments), and the LA method can recognize all of them at almost their exact boundaries. Therefore, even in the case of the most realistic scenario, the system gives 100% whole-segment recognition accuracy, while the accuracy per CSI frame detection is 90.27%. Note that, per CSI level accuracy will be affected even if the start and end of the segment is slightly missed (e.g., 2nd, 13th, 14th activity segment in Figure 11) in the prediction, while it may not be very critical in practice. Thus, through an introduction of some flexibility around borders, this accuracy can be further increased.

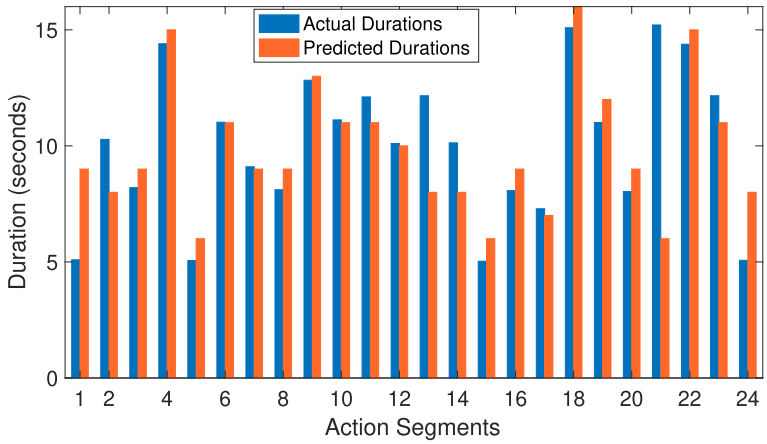


Fig. 12. Actual and predicted durations per action segment using trained parameters with D1 and applying on the validation dataset D2 (i.e., D1/D2).

Once the start and the end of a segment are known, along with the specific transmission frequency (e.g., 100 Hz in our experiments), we can also calculate the duration of that activity or non-activity segment. In other words, we can estimate the time a participant performs a given activity using the segmentation predictions. The comparison of the actual and predicted duration is shown in Figure 12, where, in general, we observe that the predicted durations align with the actual durations of the segments. Note that the proposed system can identify each individual segment in the validation dataset (as discussed in the segmentation results). However, as per the CSI frame level prediction, there are some false ones at the edges of the segments; therefore, the duration estimation is not millisecond accurate. Also note that Figure 12 shows the comparison of duration results for only the activity segments, but we also observe the same for the non-activity segments, since the CSI data collection is continuous and the non-activity segments complement the activity segments in our dataset.

4.3.2 Activity Classification. After identifying the start of a new activity segment, our system then performs classification to predict the type of the activity. As it is described in the proposed system architecture, we use a hierarchical solution for this task. That is, we begin with a binary classification to categorize the activity segment as either a wrist or finger movement. Following this, we run a corresponding model to further classify the specific wrist or finger movement within that category.

To understand the performance of this approach, we begin by looking at the confusion matrices in Figure 13(a)–(c) which represent the prediction accuracy of each model used in the hierarchical system. In Figure 13(a), the three finger activities: fist (F), OK sign (O), and victory sign (V) are clearly categorized as "finger" activities while pitch (P), roll (R), and yaw (Y) are correctly placed in the "wrist" category with accuracies all much greater than 90%. Once the category is recognized, we then look at the intra-category recognition for both finger and wrist activities separately in Figure 13(b) which achieves an accuracy of 90.02% and Figure 13(c) which achieves an accuracy of 99.17%. While there is a small amount of confusion between the predictions for the finger activities, this could be due to small pauses between each activity which can be further improved through combining predictions from proceeding time instances. Combining these three models into a hierarchical system, we achieve an overall prediction accuracy of 91.22%. The confusion matrix for this scenario is given in Figure 13(d). One key observation to be made with the use of the hierarchical approach is that if we add additional categories beyond finger and wrist (e.g., arm movements), the category-specific models do not need to be updated and as such, the prediction accuracy of these models remains the same.

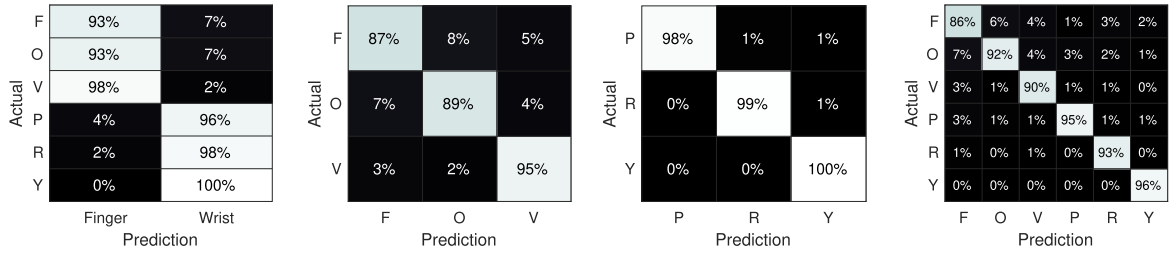


Fig. 13. (a) Confusion matrix for binary wrist-finger classification. Total accuracy: 96.44%. Confusion matrix when using an individual ML model for (b) finger, total accuracy: 90.02%, and (c) wrist, total accuracy: 99.17%. (d) Confusion matrix obtained from the entire hierarchical model. Total accuracy: 91.22%.

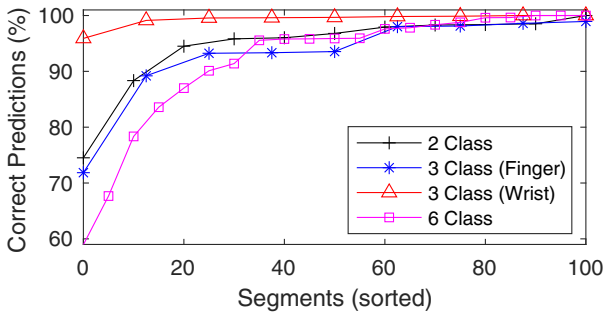


Fig. 14. Percentage of samples correctly predicted per segment.

Note that with a classical approach the activities could also be recognized through a single model, which we evaluated to have an accuracy of 89.89%. However, unlike the hierarchical approach, if we plan to add more categories (e.g., arm movements), we would need to retrain the model and due to model capacity, we would expect the per-class accuracy to suffer. In our case, we observe that fist accuracy decreases by 3%, OK sign increases by 3%, and the accuracy for the victory sign remains the same. However, if we compare the wrist activities, the six-class model decreases accuracy for all three classes, i.e., 16% for pitch, 1% for roll and 10% for yaw activities. For the yaw activity, this is particularly interesting since in the hierarchical approach, it achieves 100% prediction accuracy for both the binary wrist-finger classification phase as well as the wrist classification phase. This demonstrates that the model loses some pattern-matching capacity as the number of classes increases and demonstrates that the hierarchical approach can offer scalability when adding additional activity categories to the system.

As soon as we identify the beginning of a segment, we can immediately begin running our activity classification model pipeline on all incoming CSI samples. Obviously, not all of the activity classes achieve perfect accuracy. However, since we can also recognize the ending of the segment, we can capture statistics detailing the frequency that each class was predicted for a given segment which can then be passed through a majority voting scheme. To evaluate this, in Figure 14, we look at each of the segments in our evaluation dataset and plot the number of samples correctly classified per segment. Overall, for all of the segments, more than 50% of samples are predicted correctly, which means that majority voting would allow each segment to be predicted correctly 100% of the time for all of the evaluated segments. Of course, a higher percentage of correctly predicted samples will ensure that the model can continue to generalize on future CSI data. For example, with the six-class classifier, some segments achieve as low as 58.95% correct samples. On the other hand, the three-class wrist classifier correctly predicts more than 95% of all samples for each of the evaluated segments.

Table 5. Comparison of Different Model Sizes and Accuracies Obtained

Model	Model Size (MB)	Accuracy
2class	5.092	91.22%
finger3class	3.084	
wrist3class	8.731	
6class	3.157	89.89%

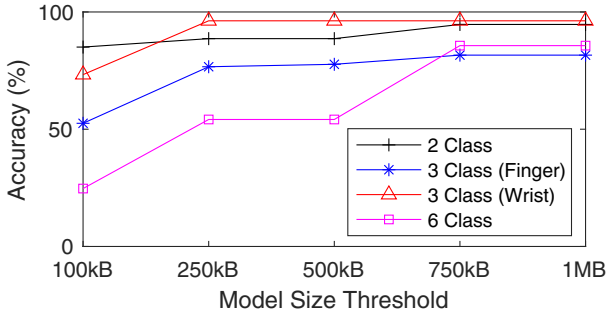


Fig. 15. Cumulative accuracy achieved when given a threshold for maximum model size.

Model Size: In identifying hyperparameters for each of the classification models, we do not directly consider the model size. As such, the model sizes shown for each model in Table 5 vary without a direct correlation to the number of classes that are predicted by the model. Furthermore, since our goal is to run our model on a low-cost embedded microcontroller system, using a consistent and smaller model size allows for a faster inference rate as well as ensures that the model can load completely in memory. As such, we apply a threshold for each of the models in Figure 15. As we increase the allowed model size threshold, we find that our optimal model hyperparameters can end up in a local-optimal value, thus we plot the cumulative maximum accuracy for each model. We can see that the six-class classifier suffers the most from decreasing the allowed model size resulting in an accuracy of 54.15% even when given 500 kB of model space compared to 77.69%, 96.23%, and 88.63% for the three-class finger classifier, three-class wrist classifier, and two-class activity-group classifier, respectively. Notice, by default, the ESP32 microcontroller we are using for CSI data collection is limited to a standard 520 kB of RAM [16]. Furthermore, the hierarchical approach allows us to automatically switch in the corresponding model into RAM as needed compared to the six-class model approach which requires that all corresponding model weights are retained in memory even if they are not important for the prediction. For example, some weights of the model will be dedicated to finger-based movements while others are dedicated to wrist-based movements. However, outside of our proposed hierarchical approach, there is no obvious approach to splitting model weights per category for a pretrained model. This demonstrates that splitting our model training using our proposed hierarchical method allows us to achieve better predictions within the memory constraints inherited from embedded machine-learning environments.

4.3.3 Activity Repetition Counting. As the next step, for each recognized activity segment, the proposed system aims to find out the number of repetitions of that activity performed within that segment. Our simple yet efficient counting method is trained for each activity type separately. It looks for repetitiveness in the dataset using the local maxima and minima on the optimized CSI time series data. Table 6 shows the MAE of counts obtained for each of the actions using the optimized parameters on the training data, i.e., the sixth column, as well as using the unforeseen validation dataset on the seventh column.

Table 6. Per Action Counting Results with Optimal Parameters (MAE 0.933 ± 0.88 for Training and 1.37 ± 0.89 for Validation)

Action	ω	ρ_s	ρ	η	Tr. Error	Val. Error
Pitch	50	25	32	7	0.95 ± 0.83	0.75 ± 0.50
Yaw	50	21	32	8	1.15 ± 0.99	2.50 ± 1.00
Roll	50	25	2	6	1.05 ± 0.94	0.75 ± 0.95
Fist	50	29	32	6	0.75 ± 0.78	2.25 ± 1.25
V-Sign	50	29	32	6	0.85 ± 0.87	1.00 ± 0.82
OK-Sign	50	32	4	6	0.85 ± 0.87	1.00 ± 0.82

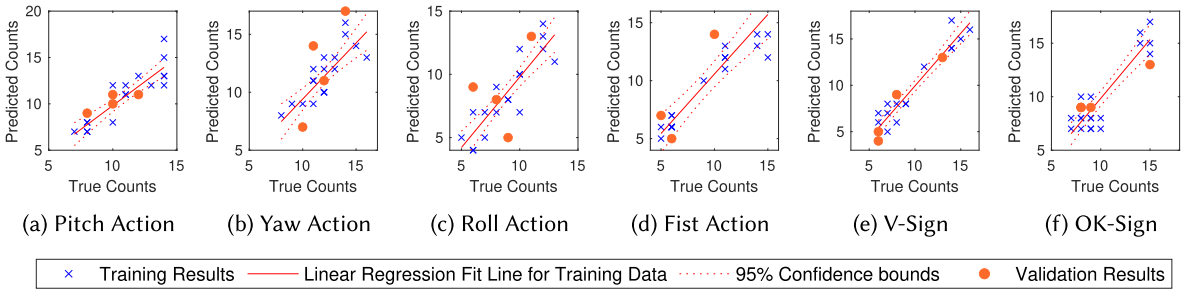
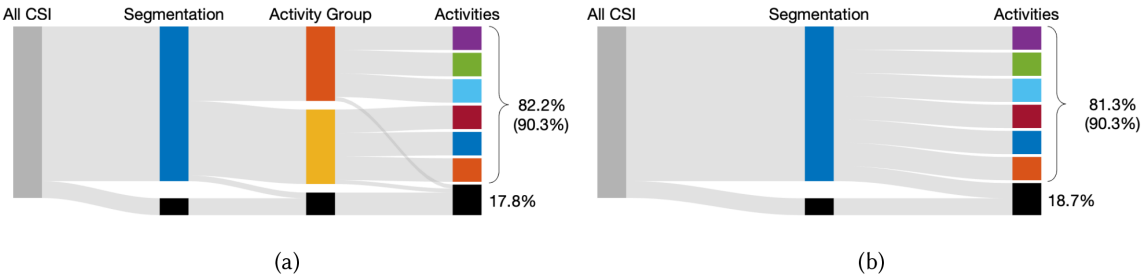


Fig. 16. Linear regression on predicted vs. true counts per activity.

First of all, the results in Table 6 show that the optimized window size ω is the same, i.e., 50 CSI frames, for each type of activity. The number of optimized repetitive averages, η, is also very similar being mostly 6, but slightly different for the pitch and yaw actions (7 and 8, respectively). Note that the number of sinusoidal peaks mainly depends on these two variables, which is the base of our proposed counting method. On the other hand, the groups of the PCs, as defined by ρ_s and ρ, are found to be completely different from those for the segmentation. The second and third PCs are found to be the most effective in identifying the action or non-action segments as already shown in Table 4. However, the PCs from 21 to 60 are found to be effective for counting the action repetitions. The group of such PCs varies for each of the six actions per the optimizer. The Equation (2) is used to calculate the MAE of the counting method per action. These errors are listed in Table 6. Linear regression lines with 95% confidence interval are plotted in Figure 16 per action counting model and show that the models do pretty well on a foreseen dataset. The confidence boundary gets thicker for the unforeseen D2 dataset using the D1 dataset's optimized parameters. It can be due to the lower number of samples in the D2 dataset. However, the validation errors are close to the training errors in the case of pitch, v-sign, and OK-sign actions, while deteriorating more in the case of yaw, roll, and fist actions.

4.4 Overall System Evaluation

So far we have evaluated the components of our proposed system independently from one another. However, since segmentation recognition, activity-group categorization, and activity-type classification tasks are calculated per CSI sample, we can calculate an overall accuracy for our proposed system. Additionally, we can identify the number of overall samples that failed any of the steps of our proposed system. We present two Sankey plots in Figure 17(a) and (b) which show the percentage of samples that pass through each state of our system when using our hierarchical model and the six-class classifier, respectively.



(a) (b)

Fig. 17. Sankey plot: (a) Hierarchical model and (b) Single six class classifier.

From Figure 17(a), we can see that there are four groups. First, we have all 100% of the collected CSI samples. Next, we have segmentation which was successful 90.27% of the time (per CSI frame accuracy in Table 4 for LA with D1/D2 case). The remaining 9.73% of samples are falsely identified as either activity or non-activity and as such, they are marked as failed for the remainder of the plot. Next, we have the binary activity-group classifier which fails 3.6% of samples. Finally, we have the activity classifier which fails around 10% and 0.83% of finger and wrist activities, respectively. Overall, 82.2% of all CSI samples successfully pass through all of the components in the system.

In the case of using the six-class classifier, as shown in Figure 17(b), we no longer separate the activity-group prediction from the actual activity prediction. The segmentation part is still successful 90.3% of the time and after classifying each of the six activities, we end up with 81.3% of activities passing successfully through our system which is lower than the number of samples that were successfully passed through with the hierarchical approach.

Notice, however, that it is not necessary for all samples to successfully pass through the system to achieve highly accurate rehabilitation exercise tracking statistics. By leveraging the segmentation start and stop points along with majority voting, we can further improve the successful throughput such that all 90.3% of the correctly segmented samples are successfully passed through the system. Specifically, while we may have a few anomalies in our predictions because we do not necessarily use the results of our predictions in real-time, we can use knowledge from the entire activity segment to inform our final predictions for all segments within the segment.

4.5 Generalizability of the Proposed System

In this part, we evaluate the generalizability and robustness of the proposed system using the other datasets we collected with different volunteers (including some having different hand conditions) and/or in different environments. We also evaluate how the proposed system performs with a new set of movements and discuss the benefit of the hierarchical design.

4.5.1 New Volunteers and Environments. As described in Tables 3 and 4, we also collected data for the same types of hand movements from ten volunteers in five different environments. The summary of the results is given in Table 7. The first row of the table shows the summary of the evaluation of the primary volunteer's actions, as elaborated in previous sections. The next rows show the results using the same optimal parameters or models from the first dataset for each of the other datasets, respectively. It can be seen from Table 7 that once the system is optimized with one volunteer's data, it works well enough for any other volunteer even if they are in different environments with the same set of parameters.

Overall, we see 94.82% segment detection accuracy, average counting error of 1.72 ± 1.19 and a classification accuracy of 83.89%. Here, note that the classification accuracy is per CSI frame, but if we consider segment-level classification, the classification accuracy reaches 97.41%. This is computed by selecting a single class (i.e., highest predicted) per segment through majority voting. As it can be seen in Figure 18, this majority voting yields 100% accuracy for action classification within each segment which makes the segmentation per CSI accuracy (S_c)

Table 7. Validation with Other Volunteer and Environment Combinations

Vol. No.	Env.	Acc. Per CSI S_c (%)	Segment Detect. S_d (%)	Duration Error (seconds)	Counting Error (MAE±Std.)	Classification Accuracy (%)				
						Binary C_b	Finger C_f	Wrist C_w	Overall C	Segment Level, C_s
1	1	90.27	100.0	1.11 ± 1.97	1.37 ± 0.89	96.44	90.02	99.17	91.22	99.15
1	1	87.53	94.48	1.35 ± 1.88	2.34 ± 1.87	94.52	86.07	98.74	87.34	98.42
2	4	84.55	93.10	1.37 ± 1.76	1.45 ± 1.33	92.17	83.56	90.12	80.04	96.92
3	1	88.43	93.10	1.42 ± 1.53	2.15 ± 1.03	94.15	86.87	97.35	86.72	98.46
4	1	81.65	94.48	2.12 ± 1.23	1.45 ± 1.11	90.63	82.11	88.36	77.25	95.83
5	1	73.62	88.97	1.82 ± 2.39	2.26 ± 1.52	84.72	81.22	87.26	71.37	92.45
5	5	90.32	100.0	1.21 ± 1.56	1.91 ± 1.33	96.11	81.41	87.65	81.24	98.18
6	1	80.17	92.42	1.89 ± 1.74	1.69 ± 1.38	98.64	84.29	90.46	86.19	97.26
7	1	88.84	98.35	1.37 ± 1.28	1.45 ± 0.88	93.24	87.45	96.89	85.94	98.43
8	1	84.52	96.69	1.64 ± 1.21	1.58 ± 1.06	95.86	88.96	98.77	89.98	98.45
9	1	83.42	93.79	1.56 ± 0.57	1.29 ± 1.05	94.63	88.06	96.34	87.25	97.89
10	4	85.64	92.41	1.54 ± 1.28	1.66 ± 0.78	92.05	86.05	92.35	82.11	97.43
Average		84.91	94.82	1.53 ± 1.53	1.72 ± 1.19	93.60	85.51	93.62	83.89	97.41

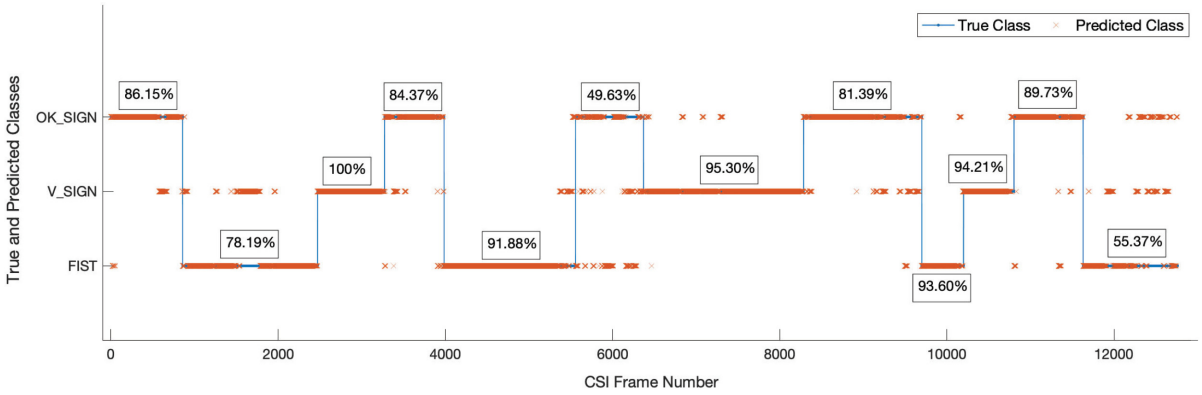


Fig. 18. Per-segment classification results showing that with majority voting each segment can be classified to one class accurately (even in the dataset that gives the worst case CSI level accuracy, i.e., volunteer 3, environment 2 in Table 8).

the actual classification accuracy for the action segments. As the predicted non-action segments can also have mis-segmented CSI for actions, we also consider the overall classification accuracy (C) for those parts. In the end, our segment-level classification accuracy can be calculated as $C_s \leftarrow S_c + C \times (100 - S_c)$. Such segment-level classification accuracy per dataset is shown in the last column of Tables 7 and 8.

While in some cases we see slightly lower results, especially for the finger and wrist action classification, this can be considered acceptable as the finger and wrist movement varies for different people but the movement detection as well as segmentation for moving vs. static environment evaluation remains the same. The fluctuation of CSI amplitudes obtained in the proposed counting system also works almost the same for all volunteers. Per activity results for all volunteers are also inline with the action-wise results given in Table 6 for the primary dataset. From all of these results, we can say that the proposed system, once optimized for the hyper-parameters with any volunteer, is generalizable for datasets collected in different environments with different volunteers.

Table 8. Validation on Special Scenarios

Vol. No.	Env.	Speciality	Acc.(%) Per CSI	Segment Det. (%)	Duration Error	Counting Error	Classification (%)	
							Overall	By Segment
1	2	Multiple people	76.27	90.34	2.73 ± 2.18	1.58 ± 2.89	68.28	92.47
1	1	Wearing gloves	90.13	99.31	1.52 ± 1.84	1.83 ± 1.63	80.98	98.12
1	1	Wearing wrist-support	88.74	97.93	1.44 ± 1.73	1.44 ± 1.57	82.44	98.02
1	3	Wearing gloves	83.78	91.72	1.94 ± 1.66	1.89 ± 1.49	82.05	97.09
2	3	Wearing gloves	77.59	91.03	1.98 ± 1.55	1.84 ± 1.55	78.99	95.29
2	3	Wearing wrist-support	79.74	92.41	1.54 ± 1.78	1.93 ± 1.19	80.11	95.97
3	2	Multiple people	74.91	85.52	1.04 ± 0.94	1.52 ± 3.04	65.04	91.23
4	1	Wearing wrist-support	87.16	95.17	1.89 ± 1.09	1.29 ± 1.53	81.03	97.56
Average			82.29	92.93	1.76 ± 1.60	1.67 ± 1.86	77.37	95.72

4.5.2 Special Scenarios. We also consider some special scenarios to test how the performance of our system is affected. To this end, we first consider some hand conditions for the users, such as wearing a wrist thumb support, as this could be the case in practice for such patients. Similarly, while in this study we assume that there is only one person[3] monitored in the physical therapy session, it is possible that there could be some other people around while the user is performing physical therapy exercises. Thus, we also consider such scenarios and analyze the effect on results. The details of these special scenarios and the results with each of them are provided in Table 8. Overall, we see a slightly lower average performance compared to the results in Table 7. As expected, this is due to the effect of either the movements of other people in the environment or due to the hand conditions of volunteers. However, the system's performance can still be considered acceptable.

4.5.3 New Set of Movements. To evaluate the scalability of our system with different sets of movements, we collected a dataset with three new movements. Complicating the movements further, we consider physical therapy exercises that include interaction with an object. These movements are (i) grip and pinch a hand therapy ball, (ii) finger spread using a hand band, and (iii) drop and grab a hand therapy ball. Note that the hierarchical design used in the classification part of the proposed system allows for developing a model for this new set of movements without changing the models for other movements. We only need to retrain the activity group classifier to recognize this new group. However, we do not need to change anything for segmentation and counting. As a result, using the same optimized parameters for segmentation and counting, we still obtain similar segmentation accuracy (84.36% per CSI and 93.10% detection accuracy) and counting error (1.79 ± 1.19). For the classification, the retrained activity group classifier for the three groups provides 96.12% accuracy, and the action classifier for this new group of three activities gives 75.57% accuracy. When we incorporate the previous two models, i.e., wrist and finger classifiers, we calculate the overall classification accuracy with all nine actions as 81.61%. With the aforementioned majority voting idea, we then obtain a segment level classification of 97.12%. This is quite close to the previous accuracy for six actions (i.e., 97.41% with two groups of activities), showing the generalizability of the proposed solution to include new movements.

4.6 Comparison to Existing Systems

Among all the existing systems, the work of Chen et al. [10] is the closest work to our study in terms of its design. While it does not provide a counting component, its segmentation results look promising. On the other hand, similar to all other datasets for segmentation studies, the dataset of this study is also for large-scale human movements, like sitting down, standing up, walking, stooping, and waving. Thus, their approach is less likely

[3]Monitoring more than one person through WiFi sensing [20] is more challenging and is out of the scope of this article.

to suit our use case of small-scale movements like finger movements. However, for the sake of comparison, we implement their approach with the Butterworth low-pass filter, PCA, and using their dynamic threshold calculation and sliding window-based segmentation methods on our small-scale activity datasets. The threshold weight parameter, α, is not properly introduced in their study, and the lengths of the two specific windows are not well specified. Thus, we run the algorithm for different possible floating values ranging from 0.1 to 0.9 for α and consider the value of 0.5 giving the best result. Besides, the threshold is described to be updated every two minutes, which led us to consider the larger window size having two minutes of CSI data frames, i.e., consisting of $2 \times 60 \times 100 = 12,000$ CSI frames of our dataset with 100 Hz transmission frequency. The shorter window is calculated to be one-sixth part of the larger window, as depicted in their segmentation figure. A major downside of their segmentation algorithm is they calculate the start points and end points of the segments disjointly, which leads to discontinuous segmentation in our dataset. Our implementation of their system can identify at best 17 starting points out of the 118 starting points of the action segments, while only nine ending points were identified, yielding a very low accuracy. Therefore, it can be deduced that their system for large-scale movement segmentation cannot be applied to our small-scale movement segmentation.

5 Conclusion

In this work, we proposed a non-intrusive, low-cost, and device-free WiFi sensing based physical rehabilitation hand activity tracking system called Wi-PT-Hand. The proposed system considers small-scale movements like wrist and finger movements, and it can not only classify an activity but also can detect when, for how long, and how many times that activity is performed using only the ubiquitous WiFi signals around us. The proposed system relies on prediction models and optimizers that can potentially run on edge devices with very low computational capabilities, thus they can be deployable at scale. The extensive experimental results with varying scenarios show that the performance of each component of the system as well as the entire system is promising and robust enough to be used in practice. Considering that the patient activity (or therapeutic dose) is currently often recorded manually by the rehabilitation specialist during treatment, there is a great potential to not only reduce therapist burden but also improve the fidelity of the data, which can then be used to develop new prediction models to improve patient outcome, reduce length of stay, and reduce overall healthcare costs. Since the proposed system only relies on WiFi CSI data and does not use any user-specific data, it provides a secure and privacy preserving solution. Moreover, the proposed system is a standalone solution, thus the collected data never leaves the device (e.g., does not go to cloud or servers), providing additional privacy and security. In our future work, we aim to look at several extensions and explore the usability of the proposed system by actual patients.

References

[1] Giovanni Abbruzzese, Roberta Marchese, Laura Avanzino, and Elisa Pelosin. 2016. Rehabilitation for Parkinson's disease: Current outlook and future challenges. *Parkinsonism & Related Disorders* 22 (2016), S60–S64.

[2] Fadel Adib, Chen-Yu Hsu, Hongzi Mao, Dina Katabi, and Frédo Durand. 2015. Capturing the human figure through a wall. *ACM Transactions on Graphics* 34, 6 (2015), 1–13.

[3] Md Atiqur Rahman Ahad, Anindya Das Antar, and Omar Shahid. 2019. Vision-based action understanding for assistive healthcare: A short review. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*. IEEE, 1–11.

[4] Sizhe An and Umit Y. Ogras. 2021. MARS: mmWave-based Assistive Rehabilitation System for Smart Healthcare. *ACM Transactions on Embedded Computing Systems* 20, 5s (2021), 1–22.

[5] Atheros CSI Tool. 2019. Retrieved from https://wands.sg/research/wifi/AtherosCSI/

[6] Min S. H. Aung, Sebastian Kaltwang, Bernardino Romera-Paredes, Brais Martínez, Aneesha Singh, Matteo Cella, Michel F. Valstar, Hongying Meng, Andrew Kemp, Moshen Shafizadeh, Aaron C. Elkins, Natalie Kanakam, Amschel de Rothschild, Nick Tyler, Paul J. Watson, Amanda C. de C. Williams, Maja Pantic, and Nadia Bianchi-Berthouze. 2016. The automatic detection of chronic pain-related expression: Requirements, challenges and the multimodal EmoPain dataset. *IEEE Transactions on Affective Computing* 7, 4 (2016), 435–451.

[7] Ganapati Bhat, Nicholas Tran, Holly Shill, and Umit Y. Ogras. 2020. w-HAR: An activity recognition dataset and framework using low-power wearable devices. *Sensors* 20, 18 (2020), 5356.

[8] Valentina Bianchi, Marco Bassoli, Gianfranco Lombardo, Paolo Fornacciari, Monica Mordonini, and Ilaria De Munari. 2019. IoT wearable sensor and deep learning: An integrated approach for personalized human activity recognition in a smart home environment. *IEEE Internet Things Journal* 6, 5 (2019), 8553–8562.

[9] Adam D. Bull. 2011. Convergence rates of efficient global optimization algorithms. arXiv:1101.3501. Retrieved from https://arxiv.org/abs/1101.3501

[10] Shiming Chen, Chunjing Xiao, Yanhui Han, and Xianghe Du. 2021. A real-time activity recognition system based on dynamic adaptive windows using WiFi signals. In *Proceedings of the 5th International Conference on Computer Science and Artificial Intelligence*, 122–127.

[11] François Chollet. 2015. Keras. Retrieved from https://keras.io

[12] Michael A. Gelbart, Jasper Snoek, and Ryan P. Adams. 2014. Bayesian optimization with unknown constraints. arXiv:1403.5607. Retrieved from https://arxiv.org/abs/1403.5607

[13] Xiaonan Guo, Jian Liu, Cong Shi, Hongbo Liu, Yingying Chen, and Mooi Choo Chuah. 2018. Device-free personalized fitness assistant using WiFi. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 2, 4 (2018), 165:1–165:23.

[14] 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.

[15] Steven M. Hernandez and Eyuphan Bulut. 2020. Lightweight and standalone IoT based WiFi sensing for active repositioning and mobility. In *Proceedings of the 21st International Symposium on "A World of Wireless, Mobile and Multimedia Networks" (WoWMoM)*, 277–286.

[16] Steven M. Hernandez and Eyuphan Bulut. 2023. WiFi sensing on the edge: Signal processing techniques and challenges for real-world systems. *IEEE Communications Surveys & Tutorials* 25, 1 (2023), 46–76.

[17] Steven M Hernandez, Deniz Erdag, and Eyuphan Bulut. 2021. Towards dense and scalable soil sensing through low-cost WiFi sensing networks. In *Proceedings of the IEEE 46th Conference on Local Computer Networks (LCN)*. IEEE, 549–556.

[18] Steven M Hernandez and Md Touhiduzzaman. 2024. CSI-Pi. Retrieved from https://github.com/MoWiNG-Lab/CSI-Pi.

[19] Steven M Hernandez, Md Touhiduzzaman, Peter E Pidcoe, and Eyuphan Bulut. 2022. Wi-PT: Wireless sensing based low-cost physical rehabilitation tracking. In *Proceedings of the IEEE International Conference on E-health Networking, Application & Services (HealthCom)*, 113–118.

[20] Jingzhi Hu, Tianyue Zheng, Zhe Chen, Hongbo Wang, and Jun Luo. 2023. MUSE-Fi: Contactless MUti-person SEnsing exploiting near-field Wi-Fi channel variation. In *Proceedings of the 29th Annual International Conference on Mobile Computing and Networking (ACM MobiCom '23)*. ACM, New York, NY, 75:1–75:15.

[21] Kyoung Bo Lee, Seong Hoon Lim, Kyung Hoon Kim, Ki Jeon Kim, Yang Rae Kim, Woo Nam Chang, Jun Woo Yeom, Young Dong Kim, and Byong Yong Hwang. 2015. Six-month functional recovery of stroke patients: A multi-time-point study. *International Journal of Rehabilitation Research* 38, 2 (2015), 173.

[22] Daniel Leightley, John Darby, Baihua Li, Jamie S McPhee, and Moi Hoon Yap. 2013. Human activity recognition for physical rehabilitation. In *Proceedings of the IEEE International Conference on Systems, Man, and Cybernetics*, 261–266.

[23] Yongsen Ma, Gang Zhou, and Shuangquan Wang. 2019. WiFi sensing with channel state information: A survey. *ACM Computing Surveys* 52, 3 (2019), 1–36.

[24] Ubaid Mehmood, Irene Moser, Prem Prakash Jayaraman, and Abhik Banerjee. 2019. Occupancy estimation using WiFi: A case study for counting passengers on busses. In *Proceedings of the IEEE 5th World Forum on Internet of Things (WF-IoT)*. IEEE, 165–170.

[25] Nexmon: The c-based Firmware Patching Framework. 2019. Retrieved from https://nexmon.org

[26] Sameera Palipana, David Rojas, Piyush Agrawal, and Dirk Pesch. 2017. FallDeFi: Ubiquitous fall detection using commodity Wi-Fi devices. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 1, 4 (2017), 155:1–155:25.

[27] Jin Park and Tae-Ho Kim. 2019. The effects of balance and gait function on quality of life of stroke patients. *NeuroRehabilitation* 44, 1 (2019), 37–41.

[28] Octavian Postolache, D. Jude Hemanth, Ricardo Alexandre, Deepak Gupta, Oana Geman, and Ashish Khanna. 2021. Remote monitoring of physical rehabilitation of stroke patients using IoT and virtual reality. *IEEE Journal of Selected Areas in Communications* 39, 2 (2021), 562–573.

[29] Jun Qi, Po Yang, Atif Waraich, Zhikun Deng, Youbing Zhao, and Yun Yang. 2018. Examining sensor-based physical activity recognition and monitoring for healthcare using Internet of Things: A systematic review. *Journal of Biomedical Informatics* 87 (2018), 138–153.

[30] Arindam Sengupta, Feng Jin, Renyuan Zhang, and Siyang Cao. 2020. mm-Pose: Real-time human skeletal posture estimation using mmWave radars and CNNs. *IEEE Sensors Journal* 20, 17 (2020), 10032–10044.

[31] Jasper Snoek, Hugo Larochelle, and Ryan P. Adams. 2012. Practical Bayesian optimization of machine learning algorithms. arXiv:1206.2944. Retrieved from https://arxiv.org/abs/1206.2944

[32] Zhiling Tang, Qianqian Liu, Minjie Wu, Wenjing Chen, and Jingwen Huang. 2021. WiFi CSI gesture recognition based on parallel LSTM-FCN deep space-time neural network. *China Communications* 18, 3 (2021), 205–215.

[33] Md Touhiduzzaman. 2024. CSI Annotator App. Retrieved from https://github.com/touhiDroid/CsiAnnotator.

[34] Connie W. Tsao, Aaron W. Aday, Zaid I Almarzooq, Alvaro Alonso, Andrea Z. Beaton, Marcio S. Bittencourt, Amelia K. Boehme, Alfred E Buxton, April P. Carson, Yvonne Commodore-Mensah, Mitchell S.V. Elkind, Kelly R. Evenson, Chete Eze-Nliam, Jane F. Ferguson,

Giuliano Generoso, Jennifer E. Ho, Rizwan Kalani, Sadiya S. Khan, Brett M. Kissela, Kristen L. Knutson, Deborah A. Levine, Tené T. Lewis, Junxiu Liu, Matthew Shane Loop, Jun Ma, Michael E. Mussolino, Sankar D. Navaneethan, Amanda Marma Perak, Remy Poudel, Mary Rezk-Hanna, Gregory A. Roth, Emily B. Schroeder, Svati H. Shah, Evan L. Thacker, Lisa B. VanWagner, Salim S. Virani, Jenifer H. Voecks, Nae-Yuh Wang, Kristine Yaffe, and Seth S. Martin. 2022. Heart disease and stroke statistics—2022 update: A report from the American Heart Association. *Circulation* 145, 8 (2022), e153–e639.

[35] Raghav H. Venkatnarayan, Shakir Mahmood, and Muhammad Shahzad. 2019. WiFi based multi-user gesture recognition. *IEEE Transactions on Mobile Computing* 20, 3 (2019), 1242–1256.

[36] Fei Wang, Yiao Gao, Bo Lan, Han Ding, Jingang Shi, and Jinsong Han. 2023. U-Shape networks are unified backbones for human action understanding from Wi-Fi signals. *IEEE Internet of Things Journal* 11, 6 (2023), 10020–10030.

[37] Fei Wang, Yunpeng Song, Jimuyang Zhang, Jinsong Han, and Dong Huang. 2019. Temporal Unet: Sample level human action recognition using wifi. arXiv:1904.11953. Retrieved from https://arxiv.org/abs/1904.11953

[38] Hao Wang, Daqing Zhang, Yasha Wang, Junyi Ma, Yuxiang Wang, and Shengjie Li. 2016. RT-Fall: A real-time and contactless fall detection system with commodity WiFi devices. *IEEE Transactions on Mobile Computing* 16, 2 (2016), 511–526.

[39] Qi Wang, Panos Markopoulos, Bin Yu, Wei Chen, and Annick Timmermans. 2017. Interactive wearable systems for upper body rehabilitation: A systematic review. *Journal of Neuroengineering and Rehabilitation* 14, 1 (2017), 1–21.

[40] Chunjing Xiao, Yue Lei, Yongsen Ma, Fan Zhou, and Zhiguang Qin. 2021. DeepSeg: Deep-learning-based activity segmentation framework for activity recognition using WiFi. *IEEE Internet of Things Journal* 8, 7 (2021), 5669–5681.

[41] Jiang Xiao, Huichuwu Li, and Hai Jin. 2022. Transtrack: Online meta-transfer learning and Otsu segmentation enabled wireless gesture tracking. *Pattern Recognition* 121 (2022), Article 108157.

[42] Xiaoqun Yu and Shuping Xiong. 2019. A dynamic time warping based algorithm to evaluate kinect-enabled home-based physical rehabilitation exercises for older people. *Sensors* 19, 13 (2019), 2882.

[43] Daqing Zhang, Hao Wang, Yasha Wang, and Junyi Ma. 2015. Anti-fall: A non-intrusive and real-time fall detector leveraging CSI from commodity WiFi devices. In *Proceedings of the 13th International Conference on Smart Homes and Health Telematics (ICOST '15)*. Springer, 181–193.

[44] Lei Zhang, Wenyuan Huang, Xiaoxia Jia, Xiaojie Fan, Xiaochen Fan, Liangyi Gong Liangyi Gong, Wenyuan Tao, and Shiwen Mao. 2022. Wi-Gym: Gymnastics activity assessment using commodity Wi-Fi. *IEEE Internet of Things Journal* 9, 23 (2022), 24051–24064. DOI: https://doi.org/10.1109/JIOT.2022.3188916

Received 31 August 2023; revised 21 April 2024; accepted 5 August 2024