

William & Mary

From the Selected Works of Yongsen Ma

Spring January 15, 2020

Improving WiFi Sensing and Networking with Channel State Information

Yongsen Ma, *William & Mary*

This work is licensed under a Creative Commons CC BY-NC-ND International License.



Available at: <https://works.bepress.com/yongsen-ma/1/>

Improving WiFi Sensing and Networking with Channel State Information

Yongsen Ma

College of William & Mary
Williamsburg, VA

Bachelor of Science, Shandong University, Jinan, Shandong, China, 2010
Master of Science, Shanghai Jiao Tong University, Shanghai, China, 2013

A Dissertation presented to the Graduate Faculty
of The College of William & Mary in Candidacy for the Degree of
Doctor of Philosophy

Department of Computer Science

College of William & Mary
January 2020

APPROVAL PAGE

This Dissertation is submitted in partial fulfillment of
the requirements for the degree of

Doctor of Philosophy



Yongsen Ma

Approved by the Committee, November 15, 2019



Committee Chair

Professor Gang Zhou, Computer Science
College of William & Mary



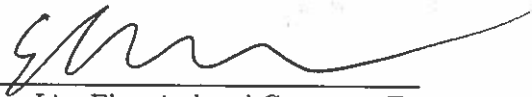
Assistant Professor Xu Liu, Computer Science
College of William & Mary



Assistant Professor Zhenming Liu, Computer Science
College of William & Mary



Assistant Professor Bin Ren, Computer Science
College of William & Mary



Assistant Professor Shan Lin, Electrical and Computer Engineering
Stony Brook University

COMPLIANCE PAGE

Research approved by

College of William & Mary Protection of Human Subjects Committee

Protocol number(s): PHSC-2017-06-28-12188-gzhou

Date(s) of approval: 07/07/2017

ABSTRACT

In recent years, WiFi has a very rapid growth due to its high throughput, high efficiency, and low costs. Multiple-Input Multiple-Output (MIMO) and Orthogonal Frequency-Division Multiplexing (OFDM) are two key technologies for providing high throughput and efficiency for WiFi systems. MIMO-OFDM provides Channel State Information (CSI) which represents the amplitude attenuation and phase shift of each transmit-receiver antenna pair of each carrier frequency. CSI helps WiFi achieve high throughput to meet the growing demands of wireless data traffic. CSI captures how wireless signals travel through the surrounding environment, so it can also be used for wireless sensing purposes. This dissertation presents how to improve WiFi sensing and networking with CSI. More specifically, this dissertation proposes deep learning models to improve the performance and capability of WiFi sensing and presents network protocols to reduce CSI feedback overhead for high efficiency WiFi networking.

For WiFi sensing, there are many wireless sensing applications using CSI as input in recent years. To get a better understanding of existing WiFi sensing technologies and future WiFi sensing trends, this dissertation presents a survey of signal processing techniques, algorithms, applications, performance results, challenges, and future trends of CSI-based WiFi sensing. CSI is widely used for gesture recognition and sign language recognition. Existing methods for WiFi-based sign language recognition have low accuracy and high costs when there are more than 200 sign gestures. The dissertation presents SignFi for sign language recognition using CSI and Convolutional Neural Networks (CNNs). SignFi provides high accuracy and low costs for run-time testing for 276 sign gestures in the lab and home environments.

For WiFi networking, although CSI provides high throughput for WiFi networks, it also introduces high overhead. WiFi transmitters need CSI feedback for transmit beamforming and rate adaptation. The size of CSI packets is very large and it grows very fast with respect to the number of antennas and channel width. CSI feedback introduces high overhead which reduces the performance and efficiency of WiFi systems, especially mobile and hand-held WiFi devices. This dissertation presents RoFi to reduce CSI feedback overhead based on the mobility status of WiFi receivers. CSI feedback compression reduces overhead, but WiFi receivers still need to send CSI feedback to the WiFi transmitter. The dissertation presents EliMO for eliminating CSI feedback without sacrificing beamforming gains.

TABLE OF CONTENTS

Acknowledgments	vi
Dedication	vii
List of Tables	viii
List of Figures	x
1 Introduction	2
1.1 Problem Statements and Contributions	3
1.1.1 Channel State Information for WiFi Sensing	3
1.1.2 Channel State Information for WiFi Networking	5
1.2 Dissertation Organization	6
2 WiFi Sensing with Channel State Information: A Survey	7
2.1 Introduction	7
2.2 Related Work	10
2.3 Background	11
2.4 Signal Processing of WiFi Sensing	14
2.4.1 Noise Reduction	14
2.4.2 Signal Transform	17
2.4.3 Signal Extraction	18
2.5 Algorithms of WiFi Sensing	22
2.5.1 Modeling-Based Algorithms	23

2.5.2	Learning-Based Algorithms	26
2.5.3	Hybrid Algorithms	29
2.6	Applications of WiFi Sensing	31
2.6.1	Detection Applications	31
2.6.2	Recognition Applications	36
2.6.3	Estimation Applications	42
2.7	Challenges and Future Trends of WiFi Sensing	47
2.7.1	Challenges for WiFi Sensing	47
2.7.2	Future WiFi Sensing Trends	49
2.7.3	Future Opportunities for Signal Processing and Algorithms of WiFi Sensing	52
2.8	Chapter Summary	53
3	SignFi: Sign Language Recognition Using WiFi	54
3.1	Introduction	54
3.2	Related Work	57
3.2.1	Sign Language Recognition	57
3.2.2	Gesture Recognition	59
3.3	Motivation	61
3.4	SignFi Design	64
3.4.1	SignFi Signal Processing	65
3.4.2	Gesture Recognition Algorithm	67
3.5	Evaluation	74
3.5.1	Experiment Setup	74
3.5.2	Comparing SignFi with Existing Methods	78
3.5.3	More Discussions on SignFi	82
3.5.4	User Independence Test	84

3.6	Chapter Summary	85
4	Location and Person Independent Activity Recognition with WiFi, Deep Neural Networks and Reinforcement Learning	86
4.1	Introduction	86
4.2	Related Work	91
4.2.1	Motion Detection with CSI	93
4.2.2	Activity Recognition with CSI	95
4.3	Design	96
4.3.1	Pre-Processing: Normalization of CSI Amplitude	97
4.3.2	Recognition Algorithm: 2D/3D CNN	99
4.3.3	State Machine: 1D CNN	101
4.3.4	Neural Architecture Search: Reinforcement Learning	104
4.4	Evaluation	107
4.4.1	Evaluation Results of Unseen Persons and Unknown Receiver Locations/Orientations	107
4.4.2	Evaluation Results of New Datasets and New Environments	114
4.5	Discussions	116
4.5.1	Overhead of Neural Architecture Search	116
4.5.2	Robustness in New Environments	117
4.6	Chapter Summary	117
5	RoFi: Rotation-Aware Channel Feedback for WiFi	119
5.1	Introduction	119
5.2	Related Work	124
5.2.1	CSI Feedback Compression	124
5.2.2	MIMO Rate Selection	124
5.2.3	Mobility-Aware WiFi Protocols	125

5.3	Motivation	126
5.3.1	Experiment Setup	126
5.3.2	Measurement Results	128
5.4	RoFi Design	131
5.4.1	RoFi Overview	131
5.4.2	Rotation Detection	132
5.4.3	Rotation-Aware Channel Feedback	135
5.4.4	Overhead Analysis	138
5.5	Evaluation	139
5.5.1	Evaluation Methodology	140
5.5.2	Performance Results of Feedback Compression	142
5.5.3	Performance Results of Rate Selection	144
5.5.4	Energy Impact of PDP Similarity Calculation	146
5.6	Chapter Summary	147
6	EliMO: Eliminating Channel State Information Feedback from MIMO	148
6.1	Introduction	148
6.2	Related Work	150
6.3	Motivation	152
6.3.1	Key Idea of EliMO	154
6.3.2	SNR Analysis	155
6.3.3	Overhead Analysis	156
6.4	EliMO Protocol Design	157
6.4.1	Frame Format	158
6.4.2	Two-Way Channel Estimation	158
6.4.3	MAC-Layer Operations	160
6.5	Evaluation	162

6.5.1	Experiment Setup	163
6.5.2	Experiment Validation	164
6.5.3	Throughput	165
6.5.4	Energy Efficiency	166
6.6	Chapter Summary	168
7	Conclusions and Future Works	169
7.1	Conclusions	169
7.2	Future Works	170

ACKNOWLEDGMENTS

I would like to thank my advisor Professor Gang Zhou for his support throughout my doctorate study. I want to thank all the members of the LENS lab for the discussions and suggestions on my research projects. I would like to thank the faculty and staff of the department of Computer Science at the College of William & Mary for creating a great environment for study and research. I also want to thank my family for their support and patience in me.

This dissertation is dedicated to my family.

LIST OF TABLES

2.1	Summary of related surveys on WiFi sensing.	10
2.2	Noise reduction techniques for WiFi sensing.	14
2.3	Signal transform techniques for WiFi sensing.	17
2.4	Signal extraction techniques for WiFi sensing.	19
2.5	Summary of WiFi sensing algorithms.	22
2.6	Pros and Cons of modeling-based and learning-based algorithms.	30
2.7	Summary of existing WiFi sensing applications.	32
2.8	Summary of WiFi sensing: detection applications.	33
2.9	Summary of WiFi sensing: recognition applications.	37
2.10	Summary of WiFi sensing: estimation applications.	43
3.1	Comparison of sign language recognition technologies.	58
3.2	Comparison of gesture recognition technologies.	60
3.3	Number of sign words in different categories used in the experiments.	76
3.4	Data collection summary.	77
3.5	CSI size of different recognition algorithms (Fig. 3.13, 3.14, and 3.15).	78
4.1	Related works of fall and motion detection with CSI.	92
4.2	Related works of activity recognition with CSI.	94
4.3	Overview of Performance Evaluation for Different Test Scenarios	107
4.4	Height and weight of experiments participants.	109
4.5	Number of parameters and inference time consumption per instance for different convolution types.	113

5.1	Existing methods for feedback compression and rate selection	140
5.2	Average energy impact of CSI feedback compression methods	146

LIST OF FIGURES

2.1	Overview of WiFi sensing and paper organization.	9
2.2	The 4D CSI tensor is a time series of CSI matrices of MIMO-OFDM channels. It captures multi-path channel variations, including amplitude attenuation and phase shifts, in spatial, frequency, and time domains.	13
2.3	Raw CSI measurements do not capture how CSI phases change over subcarriers and sampling time.	16
2.4	High pass filter for removing low-frequency signals that are reflected by static objects.	20
2.5	Thresholding of RSS and CSI amplitudes for extracting gesture signals. The user makes three sign language gestures during time 1 to 4 seconds.	20
2.6	Fresnel Zone Model. F_1 is the radius of the first Fresnel zone ($n = 1$) at point P.	24
2.7	Estimation of Angle-of-Arrival and Time-of-Flight by CSI.	24
2.8	Localization by CSIs from multiple WiFi devices and frequency bands. Real-world applications need more than three WiFi devices, assisted by clustering or majority vote, to mitigate noises and estimation errors.	26
2.9	Future trends of WiFi sensing. CSI from WiFi can be used to sense the surrounding environments, humans, animals, and objects using cross-layer information, multiple devices, and fusion of different sensors.	49

3.1	Comparison of different sign language recognition technologies. More details are shown in Table 3.1.	56
3.2	Sign words "Father" and "Mother" have the same hand gesture and finger posture, but they need the dominate hand in different locations.	62
3.3	Moving/foregrounded fingers, general and specific locations of the dominant hand of the 276 sign gestures used in our experiments. They involve more complex and diverse movements than the 25 sign gestures.	63
3.4	The average recognition accuracy of kNN with DTW decreases from 96% to 68% when the number of sign gestures increases from 25 to 276.	64
3.5	Overview of SignFi.	64
3.6	Raw CSI measurements do not capture how CSI phases change over sub-carriers and sampling time.	66
3.7	Neural architecture and parameter settings of the 9-layer CNN of SignFi.	67
3.8	An example of two-dimensional convolution with a 2×2 kernel and stride of 1, reproduced from [40].	69
3.9	Impact of batch normalization, ReLU, pooling, and dropout on recognition accuracy using leave-one-subject-out validation for 25 sign gestures and 5 users.	70
3.10	Training and testing accuracy. SignFi w/o dropout tends to overfit; there is a huge gap between training accuracy and testing accuracy.	73
3.11	Floor plan and measurement settings of the lab and home environments.	75
3.12	Experiment setup for the lab environment.	76
3.13	Recognition accuracy in different environments.	79
3.14	Recognition accuracy in different categories for the lab+home environment.	80

3.15	Time consumption of training and testing for each sign gesture. . . .	82
3.16	Impact of convolution, signal processing, and sampling rate of SignFi.	83
3.17	Recognition accuracy in different environments.	84
4.1	CSI amplitude of TX0/RX0 of the 1st subcarrier from two WiFi devices for different activities performed by two persons. Different persons or receivers have different CSI patterns. It is challenging to distinguish different activities if the recognition algorithm is trained or modeled with CSI data of receiver 1 for person 1 and tested with CSI data of receiver 2 for person 2.	88
4.2	Accuracy comparison of different deep learning solutions for WiFi-based activity recognition. The recognition accuracy of Separable-Conv2D drops dramatically for unseen persons or unknown receiver locations & orientations. The accuracy is significantly improved by the proposed design with reinforcement learning and state machine. .	89
4.3	The training process of the proposed design. The input is a time series of CSI matrices measured by WiFi packets. The recognition algorithm is a 2D CNN. The state machine is a 1D CNN. The final classification results are calculated by the concatenation of the recognition algorithm and the state machine. The neural architecture of the recognition algorithm is updated by the reinforcement learning agent by an RNN with LSTM. Motion tracking devices and audio signals are used for ground truth labeling during off-line training. During the inference stage, only CSIs are used as the input.	96

4.4	A time series of CSI amplitude measurements provides information in time, spatial, and frequency domains. The CSI tensors are fed to different CNNs to automatically find location and person independent features.	99
4.5	State machine of 5 activities in the experiments.	102
4.6	Confusion matrix of leave-one-person-out and leave-one-device-out testing results. The recognition performance for static activities, i.e., sitting and standing, is significantly improved by the state machine. .	104
4.7	Neural architecture of the best performing recognition algorithm. The input is pre-processed 4D CSI tensors, and the output is one of the five activities. The numbers inside the brackets indicate the output size of each layer.	105
4.8	Experiment setup. CSI measurements of receiver 1 are for testing and other receivers for training. There are 4 WiFi receivers placed at different locations with different heights and antenna orientations. Each participant can walk randomly within the walking area, sit or stand at two chair locations with random facing directions.	108
4.9	Average testing results of different convolution types for leave-one-person-out and leave-one-device-out validation. SeparableConv2D has the highest average score and the smallest standard deviation of recognition performance.	110
4.10	Testing results of different persons for leave-one-person-out and leave-one-device-out validation.	111
4.11	Confusion matrix of leave-one-person-out and leave-one-device-out testing results for all persons. The recognition performance for static activities, i.e., sitting and standing, is significantly improved by the state machine.	112

4.12	CSI segmentation of different activities using standard deviation of CSI amplitude. CSI data is from S.Yousefi-2017 [205].	115
4.13	Evaluation results of new datasets and dew environments: (a) test of unseen persons, (b) test of unseen environments, unseen persons, and unknown locations and orientations of the WiFi transmitter and receiver.	116
5.1	Transmit beamforming is impacted differently when the STA is in different mobility scenarios.	120
5.2	SNR results of the STA show different feedback requirements when the STA is in different mobility scenarios.	121
5.3	Some game applications need users to rotate the device. (a) Games that require device rotation. Top-left: Flight Pilot Simulator; top-right: Traffic Rider; bottom-left: Asphalt 8 Airborne; bottom-right: Bike Race. (b) Percentage of running time of each mobility type for each game.	122
5.4	Experiment setup for CSI measurements in different mobility scenarios.	126
5.5	Rotate has smaller SNR differences than MobileH and MobileV. . . .	128
5.6	Rotate has more stable SNR variations than MobileH and MobileV. .	130
5.7	RoFi design with added components in dashed rectangles.	131
5.8	Neither CSI Similarity nor Compression Noise is able to distinguish whether the STA is in the status of rotation or mobile.	133
5.9	Power Delay Profile Similarity of different mobility traces.	134
5.10	PSP is a good indicator of SNR difference and the optimal MCS selection for Rotate traces, but not for Mobile traces.	137

5.11	Normalized overhead. (a) Using fixed data rates for Rotate. (b) Statistical results for data rate of 65Mbps in different mobility scenarios. The average normalized overhead for Full Feedback is fixed at 0.82 for 65Mbps.	142
5.12	Average throughput. (a) Using fixed data rates for Rotate. (b) Statistical results for data rate of 65Mbps in different mobility scenarios.	143
5.13	Energy consumption. (a) Using fixed data rates for Rotate. (b) Statistical results for data rate of 65Mbps in different mobility scenarios.	144
5.14	Performance results of different rate selection algorithms in different mobility scenarios.	145
5.15	Energy impact of different CSI feedback schemes.	147
6.1	MAC-layer operations for implicit and explicit CSI feedback.	152
6.2	Downlink/uplink CSI estimation separately and two-way CSI estimation.	153
6.3	SNR and communication overhead analysis. EliMO provides as high SNR as explicit CSI feedback and as low overhead as implicit CSI feedback.	156
6.4	Frame format of 802.11n/ac packets with FTR and FTF.	158
6.5	Block diagram of two-way channel estimation using FTF.	159
6.6	MAC-layer operations of EliMO.	160
6.7	Flow chart of MAC-layer operations of the AP side.	162
6.8	Flow chart of MAC-layer operations of the STA side.	163
6.9	Experiment validation of SNR results.	164
6.10	Evaluation results of average throughput.	166
6.11	Evaluation results of energy consumption.	167

Improving WiFi Sensing and Networking with Channel State
Information

Chapter 1

Introduction

WiFi¹ has a very rapid growth with the increasing popularity of wireless devices. Multiple-Input Multiple-Output (MIMO) and Orthogonal Frequency-Division Multiplexing (OFDM) are two of the most important technologies for the success of WiFi. MIMO-OFDM provides Channel State Information (CSI) which represents how wireless signals propagate from transmit antennas to receive antennas at certain carrier frequencies. CSI helps WiFi achieve high throughput to meet the growing demands of wireless data traffic. CSI captures how wireless signals travel through the surrounding environment, so it can also be used for wireless sensing purposes. This dissertation presents how to improve WiFi sensing and networking with CSI.

In recent years, there are many wireless sensing applications using CSI as the input. This dissertation first presents a survey on what and how to sense with CSI. The survey gives a comprehensive review of the signal processing techniques, algorithms, applications, and performance results of WiFi sensing. WiFi sensing applications are grouped into three categories: detection, recognition, and estimation, depending on whether the outputs are binary/multi-class classifications or numerical values. The dissertation then investigates one particular WiFi sensing application in the recognition category, i.e., sign language recognition. There are many existing papers using CSI for gesture recognition. However,

¹This dissertation uses WiFi and Wi-Fi interchangeably.

existing recognition algorithms have low accuracy and high costs when there are nearly 300 sign language gestures. To address this challenge, we propose a deep learning solution for accurate and fast sign language recognition using CSI.

For WiFi networking, CSI enables beamforming to improve Signal-to-Noise Ratio (SNR) and throughput. Beamforming uses CSI to control the phase and amplitude of transmit signals for directional and/or multi-user signal transmissions. CSI can also be used for combating multi-path and frequency-selective fading effects to accurately predict Packet Delivery Ratio (PDR) and select the best transmission strategies [50, 46]. However, CSI introduces high feedback overhead that decreases the performance and efficiency of WiFi systems. It is crucial to eliminate unnecessary CSI feedback, especially for battery-powered devices. There are some existing methods on CSI feedback compression [57, 195, 162, 126], but they are not optimized for smart devices and still introduce high computation and communication costs for WiFi receivers. This dissertation first presents RoFi to reduce CSI feedback based on the mobility status of WiFi receivers. RoFi reduces 20% CSI feedback overhead for rotating WiFi receivers. RoFi requires WiFi receivers to measure CSI and calculate when to send CSI and how much CSI to send. To address this issue, the dissertation presents EliMO to eliminate CSI feedback. EliMO significantly reduces the computation and communication overhead for WiFi receivers.

This dissertation proposes deep learning models to improve the performance and capability of WiFi sensing and presents network protocols to reduce CSI feedback overhead for high efficiency WiFi networking. In the following, the dissertation presents the problem statements on improving WiFi sensing and networking with CSI. It also summarizes the contributions of each project for answering the corresponding problem statements.

1.1 Problem Statements and Contributions

1.1.1 Channel State Information for WiFi Sensing

What and how to sense with Channel State Information in literatures?

CSI represents how wireless signals propagate from the transmitter to the receiver at certain carrier frequencies. The CSI amplitude and phase are impacted by the surrounding environment and nearby humans, so CSI can be used for wireless sensing purposes. In recent years, there are many papers on CSI-based sensing applications. To get a better understanding of existing WiFi sensing applications and future WiFi sensing trends, we need to know what and how to sense with CSI in literatures.

To answer this question, §2 gives a survey on WiFi sensing with CSI. It summaries the signal processing techniques, algorithms, applications, challenges, and future trends for WiFi sensing. In summary, we make the following contributions:

- We give a comprehensive review of the basic principles, performance/cost comparisons, and best practice guidelines of the signal processing techniques and algorithms of WiFi sensing in three categories: detection, recognition, and estimation.
- We present the future trends, including cross-layer network stack, multi-device cooperation, and multi-sensor fusion, for improving the performance and efficiency of existing WiFi sensing applications and enabling new WiFi sensing opportunities.

How to recognize sign language gestures with Channel State Information?

There are many papers using CSI to recognize hand [118, 2, 100, 144, 154, 136] and finger [79, 146, 219, 100] gestures. WiFi signals are used to recognize ASL gestures in [136, 79, 100], but they are only evaluated on simple ASL gestures: 5 hand gestures in [136], 9 finger postures in [79], and 25 hand/finger gestures in [100]. The problem is how to recognize nearly 300 basic sign gestures that are frequently used in daily life.

§3 presents SignFi to answer this question. We first show that existing recognition algorithms, e.g., k-Nearest Neighbor (kNN) and Dynamic Time Wrapping (DTW), have low accuracy and high costs when there are nearly 300 sign gestures. We propose SignFi with a 9-layer Convolutional Neural Network (CNN) to provide high accuracy and low inference costs. In summary, we make the following contributions:

- We propose a new signal processing technique to remove noises from raw CSI measurements. The information of CSI change patterns is recovered.
- We present a 9-layer Convolutional Neural Network for accurate and fast sign gesture recognition using WiFi signals.
- The accuracy of SignFi is 94% for 8,280 instances of 276 sign gestures from lab and home environments and 86.66% for 7,500 instances of 150 sign gestures from 5 users.

1.1.2 Channel State Information for WiFi Networking

How to reduce Channel State Information feedback for WiFi?

CSI provides high SNR and throughput for WiFi networks, but it also introduces high overhead, especially for mobile and handheld devices. WiFi transmitters need CSI feedback to calculate the beamforming matrix and select the best transmission strategies. The transmission time for data packets is dramatically sacrificed for sending CSI and control packets. So the problem is how to reduce CSI feedback for WiFi devices.

§5 presents RoFi, rotation-aware WiFi channel feedback, to reduce CSI feedback for WiFi receivers. RoFi recognizes the mobility status of WiFi receivers and sends CSI feedback only when it is necessary. In summary, we make the following contributions:

- We run experiments to show that WiFi transmitters have different CSI feedback requirements when WiFi receivers are in different mobility statuses.
- We show the failure of existing metrics in distinguishing rotation from other mobility scenarios. We propose a new metric to detect the mobility status of WiFi receivers.
- We present rotation-aware CSI feedback to reduce CSI feedback with negligible SNR decrease. It improves the performance and efficiency of WiFi receivers.

How to eliminate Channel State Information feedback for WiFi?

There are some existing methods on reducing CSI feedback overhead [57, 195, 162, 126], but they are not optimized for smart devices and still introduce high computation

and communication costs for MIMO receivers. WiFi transmitters can use implicit CSI feedback, which uses the transpose of uplink CSI as the downlink CSI, to reduce feedback overhead [61]. But it has very low beamforming gains, since real-world MIMO channels are not reciprocal due to baseband-to-baseband channels and interferences [61, 114]. So the problem is how to eliminate CSI feedback without sacrificing beamforming gains.

Chapter 6 presents EliMO to address this issue. It uses two-way CSI estimation to accurately estimate downlink CSI without explicit CSI feedback. Based on theoretical analysis and experiment measurements, EliMO provides as low overhead as implicit CSI feedback and as high SNR as explicit CSI feedback. It significantly reduces the computation and communication overhead for MIMO receivers. In summary, we make the following contributions.

- We present two-way channel estimation allowing the AP to accurately estimate downlink CSI without explicit CSI feedback.
- We propose Feedback Training Field to completely eliminate CSI feedback without sacrificing beamforming gains.

1.2 Dissertation Organization

The rest of the dissertation is organized as follows. For WiFi sensing, §2 presents a survey on WiFi sensing in terms of signals processing techniques, algorithms, applications, challenges, and future trends, §3 presents a sign language recognition design using WiFi CSI and deep neural networks, and §4 presents a deep learning solution with neural networks and reinforcement learning for person and location independent activity recognition with WiFi. For WiFi networking, §5 presents how to reduce CSI feedback based on the mobility status of WiFi receivers, and §6 presents how to eliminate CSI feedback for WiFi receivers by estimating both downlink and uplink CSI at WiFi transmitters. Finally, §7 presents future works and concludes the dissertation.

Chapter 2

WiFi Sensing with Channel State Information: A Survey

2.1 Introduction

WiFi has a very rapid growth with the increasing popularity of wireless devices. One important technology for the success of WiFi is Multiple-Input Multiple-Output (MIMO), which provides high throughput to meet the growing demands of wireless data traffic. Along with Orthogonal Frequency-Division Multiplexing (OFDM), MIMO provides Channel State Information (CSI) for each transmit and receive antenna pair at each carrier frequency. Recently, CSI measurements from WiFi systems are used for different sensing purposes. WiFi sensing reuses the infrastructure that is used for wireless communication, so it is easy to deploy and has low cost. Moreover, unlike sensor-based and video-based solutions, WiFi sensing is not intrusive or sensitive to lighting conditions.

CSI represents how wireless signals propagate from the transmitter to the receiver at certain carrier frequencies along multiple paths. For a WiFi system with MIMO-OFDM, CSI is a 3D matrix of complex values representing the amplitude attenuation and phase shift of multi-path WiFi channels. A time series of CSI measurements captures how wireless signals travel through surrounding objects and humans in time, fre-

quency, and spatial domains, so it can be used for different wireless sensing applications. For example, CSI amplitude variations in the time domain have different patterns for different humans, activities, gestures, etc., which can be used for human presence detection [3, 230, 193, 229, 121, 119, 233, 184, 109, 140, 186, 39], fall detection [157, 110, 52, 214, 216], motion detection [42, 87, 38, 81, 202], activity recognition [6, 23, 27, 30, 33, 43, 158, 165, 166, 173, 170, 176, 189, 211, 25, 192, 29, 96], gesture recognition [123, 203, 213, 2, 3, 53, 118, 146, 154, 219, 228, 95, 79, 100, 136, 4, 5, 78, 80], and human identification/authentication [16, 17, 54, 164, 198, 212, 218, 85, 86, 137, 163, 190]. CSI phase shifts in the spatial and frequency domains, i.e., transmit/receive antennas and carrier frequencies, are related to signal transmission delay and direction, which can be used for human localization and tracking [229, 159, 82, 120, 122, 178, 112, 58, 154, 207, 68, 72, 187, 215, 96, 164, 202, 208, 144, 216]. CSI phase shifts in the time domain may have different dominant frequency components which can be used to estimate breathing rate [217, 93, 1, 161, 169, 90]. Different WiFi sensing applications have their specific requirements of signal processing techniques and classification/estimation algorithms. To get a better understanding of existing WiFi sensing technologies and gain insights into future WiFi sensing directions, this survey gives a review of the signal processing techniques, algorithms, applications, performance results, challenges, and future trends of WiFi sensing with CSI.

The overview of the survey is shown in Fig. 2.1. The background of CSI, including mathematical models, measurement procedures, real-world WiFi models, basic processing principles, and experiment platforms, is presented in §2.3. Raw CSI measurements are fed to the signal processing module for noise reduction, signal transform, and/or signal extraction, as shown in §2.4. Pre-processed CSI traces are fed to modeling-based, learning-based, or hybrid algorithms to get the output for different WiFi sensing purposes, as shown in §2.5. Depending on the output types, WiFi sensing can be grouped into three categories: detection/recognition applications try to solve binary/multi-class classification problems, and estimation applications try to get the quantity values of different tasks. §2.6

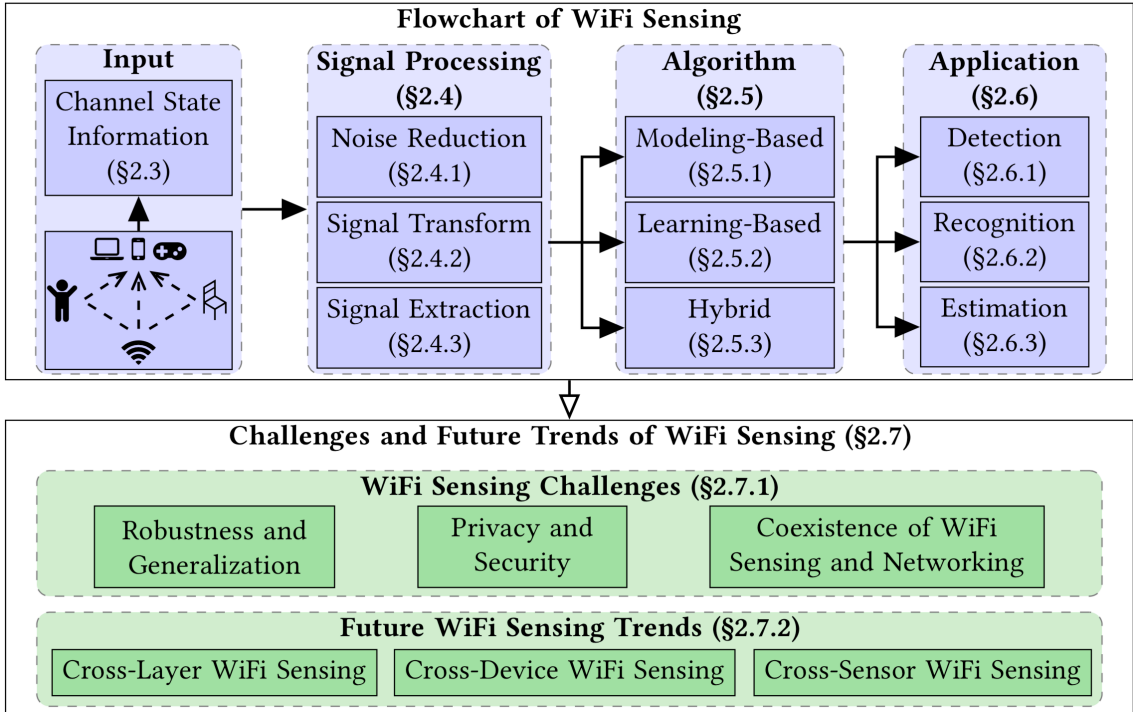


Figure 2.1: Overview of WiFi sensing and paper organization.

summaries and compares the signal processing techniques, algorithms, output types, and performance results of different WiFi sensing applications. With the development and deployment of new WiFi systems, there will be more WiFi sensing opportunities. §2.7 gives the future trends and challenges for enhancing existing WiFi sensing capabilities and enabling new WiFi sensing purposes. In summary, we make the following contributions:

- We give a comprehensive review of the basic principles, performance/cost comparisons, and best practice guidelines of the signal processing techniques and algorithms of WiFi sensing in three categories: detection, recognition, and estimation.
- We present the future trends, including cross-layer network stack, multi-device cooperation, and multi-sensor fusion, for improving the performance and efficiency of existing WiFi sensing applications and enabling new WiFi sensing opportunities.

2.2 Related Work

Table 2.1: Summary of related surveys on WiFi sensing.

Reference	Application Scope	Topic Focus
E. Wengrowski [180]	device-free localization, pose estimation, fall detection	approaches: Line-of-Sight sensors, Radio Tomographic Imaging, Through-wall RF tracking
J. Xiao et al. [194]	device-free and device-based indoor localization	models, basic principles, and data fusion techniques
Z. Yang et al. [204]	CSI-based and RSSI-based localization	basic principles and future trends; differences between CSI-based and RSSI-based solutions
S.-K. Kim [73]	motion recognition and human identification	big data analysis
D. Wu et al. [186]	human sensing	pattern-based and model-based approaches
Y. Zou et al. [238]	human behavior recognition	data-driven and model-based approaches
Z. Wang et al. [174]	human behavior recognition	basics and applications
S. Yousefi et al. [205]	human behavior recognition	deep learning techniques
This survey	All the above applications and other detection, recognition, and estimation applications	signal processing techniques, modeling-/learning-based algorithms, applications, performance results, challenges, future trends

There are some surveys on specific types of WiFi sensing applications, including localization [194, 204, 180], gesture recognition [180], and activity recognition [238, 174, 180, 73, 186, 205]. In [180], the author explores device-free human localization using wireless signal reflections; the survey also discusses device-free pose estimation and fall detection. Xiao et al. [194] give a survey on both device-free and device-based indoor localization using wireless signals; the survey focuses on the models, basic principles, and data fusion techniques. Yang et al. [204] present a survey on CSI-based localization with an emphasis on the basic principles and future trends; the survey also highlights the differences between CSI and RSSI in terms of network layering, time resolution, frequency resolution, stability, and accessibility. In [73], the author gives a brief review on human

motion recognition and human identification using CSI and big data analysis. Each of the four papers [238, 174, 186, 205] gives a survey on CSI-based human behavior recognition with their specific emphasis: basics and applications [174], deep learning techniques [205], data-driven and model-based approaches [238], and pattern-based and model-based approaches [186].

This survey is different from existing ones in that its scope is not limited to any specific type of WiFi sensing applications, as summarized in Table 2.1. The application scope of this survey includes but is not limited to human detection, motion detection, activity recognition, gesture recognition, human tracking, respiration estimation, human counting, and sleeping monitoring. The survey gives a comprehensive summary and comparison of the signal processing techniques, algorithms, and performance results of a wide variety of WiFi sensing applications. Signal processing techniques are classified into three groups: noise reduction, signal transform, and signal extraction. WiFi sensing algorithms are grouped into modeling-based and learning-based algorithms with their specific advantages and limitations. It also gives a guidance of how to select the algorithms and the corresponding signal processing techniques for different WiFi sensing applications. Finally, the survey presents future trends and challenges for enhancing existing WiFi sensing capabilities and enabling new WiFi sensing opportunities.

2.3 Background

CSI characterizes how wireless signals propagate from the transmitter to the receiver at certain carrier frequencies. CSI amplitude and phase are impacted by multi-path effects including amplitude attenuation and phase shift. Each CSI entry represents the Channel Frequency Response (CFR)

$$H(f; t) = \sum_n^N a_n(t) e^{-j2\pi f\tau_n(t)}, \quad (2.1)$$

where $a_i(t)$ is the amplitude attenuation factor, $\tau_i(t)$ is the propagation delay, and f is the carrier frequency [148]. The CSI amplitude $|H|$ and phase $\angle H$ are impacted by the displacements and movements of the transmitter, receiver, and surrounding objects and humans. In other words, CSI captures the wireless characteristics of the nearby environment. These characteristics, assisted by mathematical modeling or machine learning algorithms, can be used for different sensing applications. This is the rationale for why CSI can be used for WiFi sensing.

A WiFi channel with MIMO is divided into multiple subcarriers by OFDM. To measure CSI, the WiFi transmitter sends Long Training Symbols (LTFs), which contain pre-defined symbols for each subcarrier, in the packet preamble. When LTFs are received, the WiFi receiver estimates the CSI matrix using the received signals and the original LTFs. For each subcarrier, the WiFi channel is modeled by $\mathbf{y} = H\mathbf{x} + \mathbf{n}$, where \mathbf{y} is the received signal, \mathbf{x} is the transmitted signal, H is the CSI matrix, and \mathbf{n} is the noise vector. The receiver estimates the CSI matrix H using the pre-defined signal \mathbf{x} and received signal \mathbf{y} after receive processing such as removing cyclic prefix, demapping, and OFDM demodulation. The estimated CSI is a three dimensional matrix of complex values.

In real-world WiFi systems, the measured CSI is impacted by multi-path channels, transmit/receive processing, and hardware/software errors. The measured baseband-to-baseband CSI is

$$H_{i,j,k} = \underbrace{\left(\sum_n^N a_n e^{-j2\pi d_{i,j,n} f_k / c} \right)}_{\text{Multi-Path Channel}} \underbrace{e^{-j2\pi \tau_i f_k}}_{\text{Cyclic Shift Diversity}} \underbrace{e^{-j2\pi \rho f_k}}_{\text{Sampling Time Offset}} \underbrace{e^{-j2\pi \eta (f'_k / f_k - 1) f_k}}_{\text{Sampling Frequency Offset}} \underbrace{q_{i,j} e^{-j2\pi \zeta_{i,j}}}_{\text{Beamforming}}, \quad (2.2)$$

where $d_{i,j,n}$ is the path length from the i -th transmit antenna to the j -th receive antenna of the n -th path, f_k is the carrier frequency, τ_i is the time delay from Cyclic Shift Diversity (CSD) of the i -th transmit antenna, ρ is the Sampling Time Offset (STO), η is the Sampling Frequency Offset (SFO), and $q_{i,j}$ and $\zeta_{i,j}$ are the amplitude attenuation

and phase shift of the beamforming matrix. WiFi sensing applications need to extract the multi-path channel that contains the information of how the surrounding environment changes. Therefore, signal processing techniques are needed to remove the impact of CSD, STO, SFO, and beamforming, which is introduced in §2.4.

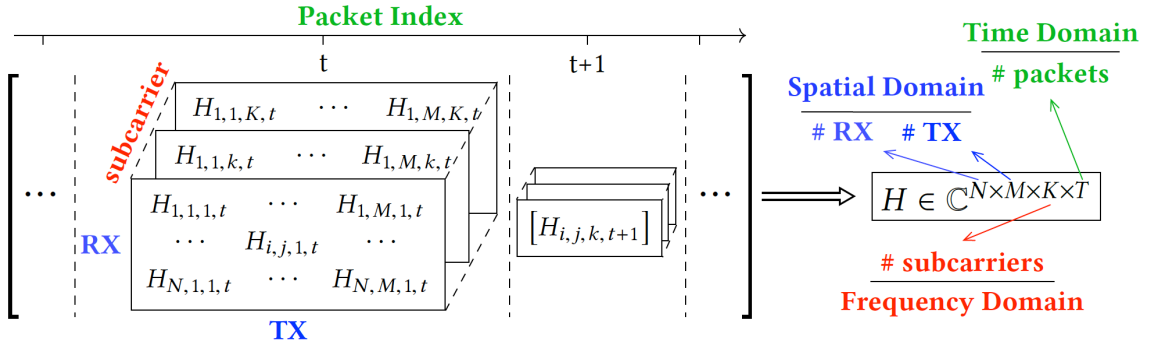


Figure 2.2: The 4D CSI tensor is a time series of CSI matrices of MIMO-OFDM channels. It captures multi-path channel variations, including amplitude attenuation and phase shifts, in spatial, frequency, and time domains.

A time series of CSI matrices characterizes MIMO channel variations in time, frequency, and spatial domains, as shown in Fig. 2.2. For a MIMO-OFDM channel with M transmit antennas, N receive antennas, and K subcarriers, the CSI matrix is a 3D matrix $H \in \mathbb{C}^{N \times M \times K}$ representing amplitude attenuation and phase shift of multi-path channels. CSI provides more information than Received Signal Strength Indicator (RSSI). The 3D CSI matrix is similar to a digital image with spatial resolution of $N \times M$ and K color channels, so CSI-based WiFi sensing can reuse the signal processing techniques and algorithms designed for computer vision tasks. The 4D CSI tensor provides additional information in the time domain. CSI can be processed, modeled, and trained in different domains for different WiFi sensing purposes, e.g., detection, recognition, and estimation.

Although CSI is included in WiFi since IEEE 802.11n, it is not reported by all off-the-shelf WiFi cards. The 802.11n CSI Tool [51] is the most widely used tool for CSI measurements. It uses Intel 5300 WiFi cards to report compressed CSIs by 802.11n-compatible WiFi networks. It provides C scripts and MATLAB source code for CSI measurements and processing. OpenRF [76] is a similar tool modified based on the 802.11n

CSI Tool. The Atheros CSI Tool [196] gives uncompressed CSIs using Qualcomm Atheros WiFi cards. For a 20MHz WiFi channel, the number of CSI subcarriers is 52 for the Atheros CSI Tool and 30 for the 802.11n CSI Tool. Both 802.11n CSI Tool and Atheros CSI Tool can operate at 2.4GHz and 5GHz. Software Defined Radio (SDR) platforms, such as Universal Software Radio Peripheral (USRP) [149] and Wireless Open Access Research Platform (WARP) [129], provide CSI measurements at 2.4GHz, 5GHz, and 60GHz.

2.4 Signal Processing of WiFi Sensing

This section presents signal processing techniques, including noise reduction, signal transform, and signal extraction, for WiFi sensing.

2.4.1 Noise Reduction

Raw CSI measurements contain noises and outliers that could reduce WiFi sensing performance. Table 2.2 gives a summary of noise reduction techniques for WiFi sensing.

Table 2.2: Noise reduction techniques for WiFi sensing.

Phase Offsets Removal	Removing phase offsets, e.g., Sampling Time/Frequency Offset, Carrier Frequency Offset, Cross-Device Synchronization Errors, Packet Detection Delay, etc., by phase difference [87, 81, 188, 192, 169, 168, 44] and (multiple) linear regression [75, 95].
Outliers Removal	Removing outliers and noises by Moving Average [193, 52, 7, 113, 43, 219, 79, 16, 207, 88, 93, 156], Median Filter [216, 227, 158, 192, 136, 17, 135], Low Pass Filter [4, 5, 183, 170, 29, 192, 96, 79, 100, 136, 135, 17], Wavelet Filter [89, 90, 161, 233, 110, 189, 203, 2, 53, 146], Hampel Filter [121, 119, 184, 233, 89, 90, 113, 222, 223, 66, 79, 16, 88, 93, 156, 168, 169], Local Outlier Factor [52, 113, 173, 203, 53], Signal Nulling [3, 32, 188, 68, 55], etc.

Phase Offsets Removal. In real-world WiFi systems, raw CSI measurements contain phase offsets due to hardware and software errors. For example, Sampling Time/Frequency Offsets (STO/SFO) are due to unsynchronized sampling clocks/frequencies of the receiver

and transmitter. Some detection and recognition applications are not very sensitive to phase offsets. It is more important to get CSI change patterns. A simple solution is to use CSI phase differences of adjacent time samples or subcarriers. It cancels CSI phase offsets with the assumption that phase offsets are the same across packets and subcarriers. It does not give accurate phases but can recover phase change patterns which can be fed to classification algorithms.

Many estimation applications require accurate phase shifts. Phase offsets introduce estimation errors for Angle-of-Arrival (AoA) and Time-of-Flight (ToF), which are used to track and localize humans and objects. SpotFi [75] removes STO/SFO by linear regression, but it does not consider different phase shifts of different transmit antennas due to CSD. This is addressed by multiple linear regression proposed in SignFi [95]. From equation (2.2), the measured CSI phase is

$$\Theta_{i,j,k} = \Phi_{i,j,k} + 2\pi f_{\delta} k (\tau_i + \rho + \eta (f'_k / f_k - 1)) + 2\pi \zeta_{i,j}, \quad (2.3)$$

where $\Phi_{i,j,k}$ is the CSI phase caused by multi-path effects, τ_i , ρ , η , and $\zeta_{i,j}$ are the phase offsets caused by CSD, STO, SFO, and beamforming, respectively, and f_{δ} is the frequency spacing of two consecutive subcarriers. The phase offsets are estimated by minimizing the fitting errors across K subcarriers, N transmit antennas, and M receive antennas

$$\hat{\tau}, \hat{\omega}, \hat{\beta} = \arg \min_{\tau, \omega, \beta} \sum_{i,j,k} (\Theta_{i,j,k} + 2\pi f_{\delta} k (i\tau + \omega) + \beta)^2, \quad (2.4)$$

where η , ω and β are the curve fitting variables [95]. As shown in Fig. 2.3a, the unwrapped CSI phases of each transmit antenna have different slopes caused by CSD. Pre-processed CSI phases $\hat{\Phi}_{i,j,k}$ are obtained by removing the estimated phase offsets, $\hat{\tau}$, $\hat{\omega}$, $\hat{\beta}$, from the measured CSI phases $\Theta_{i,j,k}$.

Phase offset removal also improves performance for binary and multi-class classification applications. It recovers CSI phase patterns over subcarriers and sampling time. The

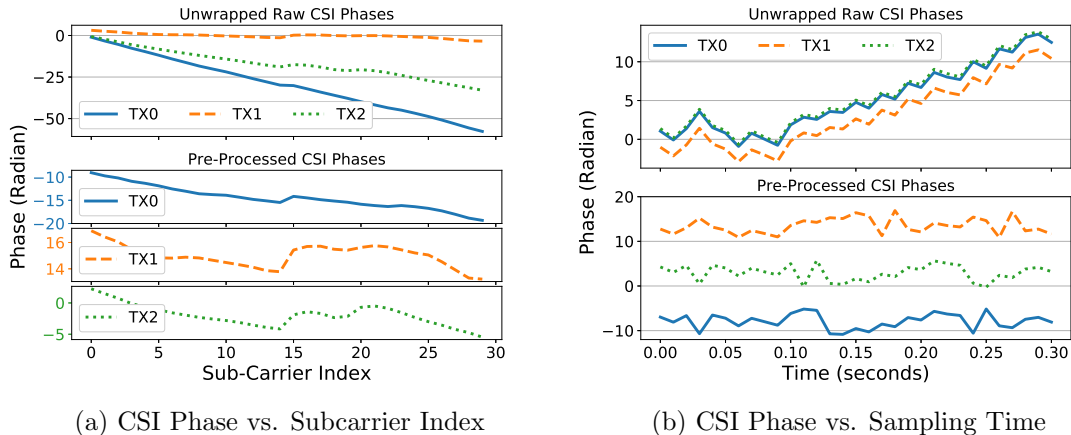


Figure 2.3: Raw CSI measurements do not capture how CSI phases change over subcarriers and sampling time.

raw measured CSI phases give redundant information about how CSI phases change. Phase offset removal unwraps CSI phases and recovers the lost information. As shown in Fig. 2.3a, raw CSI phases change periodically from $-\pi$ to π , while pre-processed CSI phases change nearly linearly in a wider range. CSI phase variations over time are also corrected. As shown in Fig. 2.3b, raw CSI phases of the first and second transmitting antenna change similarly, but they have very different patterns after pre-processing.

Outliers Removal. Moving Average and Median Filters are simple and widely used methods to remove high frequency noises. Each data point is replaced by the average or median of neighboring data points. Usually a sliding window and multiplying factors are used to give different weights, e.g., Weighted Moving Average (WMA) and Exponentially Weighted Moving Average (EWMA). Low Pass Filters (LPF) can also remove high frequency noises assisted by signal transform methods, e.g., Fast Fourier Transform (FFT). Wavelet Filter is similar to LPFs; the major difference is that it uses Discrete Wavelet Transform (DWT) instead of FFT. Details of signal transform methods and frequency-domain filters are introduced in §2.4.2 and 2.4.3.

The Hampel Filter computes the median m_i and standard deviation σ_i of a window of nearby data points. If $|x_i - m_i|/\sigma_i$ is larger than a given threshold, the current point

x_i is identified as an outlier and replaced with the median m_i . Sometimes the outliers are dropped rather than being replaced by the medians. Local Outlier Factor (LOF) is widely used in anomaly detection. It measures the local density of a given data point with respect to its neighbors. The local density is calculated by the reachability distance from a certain point to its neighbors. The data points with a significantly lower local density than their neighbors are marked as outliers. Signal Nulling is a special method for WiFi sensing to remove outliers. WiFi devices can use hardware, e.g., directional antennas, and software, e.g., transmit beamforming, algorithms for canceling noise signals.

2.4.2 Signal Transform

Signal transform methods are used for time-frequency analysis of a time series of CSI measurements. Note that the signal transform output in this scope represents the frequency of CSI change patterns rather than the carrier frequency. The summary of signal transform methods is shown in Table 2.3.

Table 2.3: Signal transform techniques for WiFi sensing.

Fast Fourier Transform	$X[k] = \sum_{n=1}^N x[n]e^{-j2\pi kn/N}$; k : frequency index. [202, 66, 158, 192, 2, 118, 219, 136, 16, 212, 137, 187, 1, 27, 88, 168, 44, 55]
Short Time Fourier Transform	$X(t, k) = \sum_{n=-\infty}^{\infty} x[n]w[n-t]e^{-jkn}$; t : time index, k : frequency index, w : window function. [110, 157, 227, 203, 123, 153, 16, 164, 122, 120, 208]
Discrete Hilbert Transform	$H[\omega] = X[\omega] \cdot (-j \cdot \text{sgn}(\omega))$; ω : frequency index, $X[\cdot]$: Fast Fourier Transform, $\text{sgn}(\cdot)$: sign function. [227, 207]
Discrete Wavelet Transform	approximation coefficients: $y_{1,low}[n] = \downarrow Q[\sum_{k=-\infty}^{\infty} x[k]g[n-k]]$, detail coefficients: $y_{1,high}[n] = \downarrow Q[\sum_{k=-\infty}^{\infty} x[k]h[n-k]]$; $\downarrow Q[\cdot]$: downsampling filter, $g[n]$: low pass filter, $h[n]$: high pass filter. [233, 110, 202, 90, 4, 5, 165, 166, 189, 203, 155, 2, 146, 154, 79, 78, 80, 198, 202, 1, 89, 161, 168]

FFT is widely used to find the distinct dominant frequencies and can be combined with a LPF to remove high frequency noises. It can also get the target signals in certain

frequencies with Band Pass Filters (BPF). For example, a time series of CSIs has different dominant frequencies when a nearby person is static or moving. FFT and BPFs can be used for human motion detection and breathing estimation, as shown in §2.4.3. Short-Time Fourier Transform (STFT) divides the input into shorter segments of equal length and computes the FFT coefficients separately on each segment, as shown in Table 2.3. STFT can identify the change of dominant frequencies over time by representing the time series data in both time and frequency domains. DHT adds an additional phase shift of $\pi/2$ to the negative frequency components of FFT, as shown in Table 2.3. It converts a time series of real-valued data to its analytic representation, i.e., a complex helical sequence. DHT is useful for analyzing the instantaneous attributes of a time series of CSI measurements.

STFT has no guarantee of good frequency resolution and time resolution simultaneously. A long window length gives good frequency resolution but poor time resolution. The frequency components can be easily identified but the time of frequency changes cannot be located. On the other hand, a short window length allows to detect when the signals change but cannot precisely identify the frequencies of the input signals. Wavelet Transform gives both good frequency resolution for low-frequency signals and good time resolution for high-frequency signals. The output of DWT can be fed to a wavelet filter to remove noises. DWT preserves mobility information in different scenarios and is more robust than Doppler phase shift [165, 166].

2.4.3 Signal Extraction

Signal extraction is for extracting target signals from raw or pre-processed CSI measurements. Sometimes it needs thresholding, filtering, or signal compression to remove unrelated or redundant signals. In some cases, it requires composition of multiple signal sources and data interpolation to get more information. Table 2.4 shows signal extraction methods widely used for WiFi sensing.

Filtering and Thresholding. High, low, and band pass filters are widely used

Table 2.4: Signal extraction techniques for WiFi sensing.

Thresholding and Filtering	Excluding signals with certain frequencies, power levels, etc., by filtering [214, 42, 30, 6, 27, 43, 158, 78, 80, 16, 164, 198, 44, 157, 81, 202, 227, 211, 123, 177, 118, 228, 137, 122, 120, 1, 88, 135] or thresholding [110, 157, 216, 42, 7, 85, 86, 222, 223, 30, 185, 231, 27, 43, 66, 165, 166, 170, 192, 123, 2, 153, 154, 219, 78, 80, 16, 164, 198, 159, 82, 68, 207, 187, 144, 178, 1, 88, 135, 156, 161, 168, 169, 172, 171, 235]; separating signals into different domains, e.g., direct/reflected paths and LoS/NLoS paths [82, 178].
Signal Compression	Removing unrelated/redundant signals by dimension reduction such as PCA [109, 229, 233, 110, 202, 4, 5, 32, 113, 227, 232, 27, 165, 166, 29, 192, 123, 146, 153, 154, 79, 78, 80, 164, 198, 229, 122, 120, 207], ICA [108, 54], SVD [89, 90, 32, 190], etc., or metrics such as self/cross correlation [39, 222, 223, 66, 190, 187, 184, 144], Euclidean distance [24, 42, 7, 67, 188], distribution function [27], etc.
Signal Composition	Composition of signals from multiple devices [75, 89, 90, 170, 211, 203, 136, 144, 92, 161, 191, 55], carrier frequencies [215, 150, 196], etc.

to extract signals with certain dominant frequencies. For example, the average resting respiration rates of adults are from 12 to 18 breaths per minute. WiFi-based respiration monitoring can use a BPF to capture the impact of chest movements caused by inhalation and exhalation. It can also filter out high-frequency components caused by motions. The input signals for filtering are usually from FFT, DHT, or STFT. Butterworth pass filters are widely used due to its monotonic amplitude response in both passband and stopband and quick roll-off around the cutoff frequency. High Pass Filters (HPFs) can be used to filter out signals from static objects that have relatively stable signal reflections. WiFi-based gesture recognition can use HPF to extract the target signals reflected by human movements, as shown in Fig. 2.4. Combined with DWT, wavelet filters are also used for outliers removal.

In the time domain, thresholding can be used to extract signals with certain power levels, AoAs, ToFs, etc. As shown in equation (2.1), CSI is impacted by wireless signals

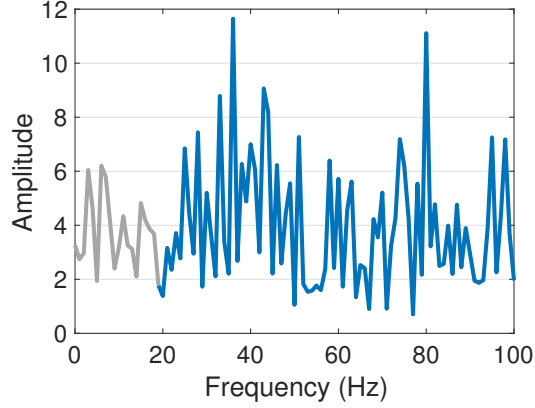


Figure 2.4: High pass filter for removing low-frequency signals that are reflected by static objects.

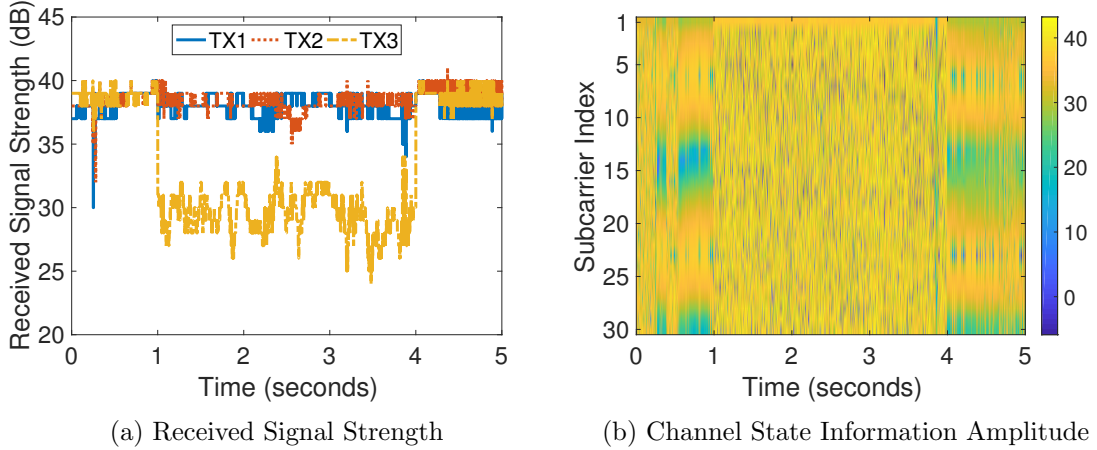


Figure 2.5: Thresholding of RSS and CSI amplitudes for extracting gesture signals. The user makes three sign language gestures during time 1 to 4 seconds.

from multi-path channels. Device-free human tracking can exclude signals of the direct path by cutting off signals with the shortest ToF. The ToFs of different paths can be calculated by Power Delay Profile (PDP), which is shown in §2.5.1. WiFi-based gesture recognition can use thresholding to exclude signals when the user is not making gestures. As shown in Fig. 2.5a, when the user is making gestures, the RSS of TX3 are higher than that when the user is static. The CSI amplitudes are also in different ranges when the user is making gestures, as shown in Fig. 2.5b. Thresholding of other metrics, e.g., CSI cross correlation, can be used for signal compression.

Signal Compression. Processing raw CSI measurements sometimes requires extensive computation resources. For example, $size(H) = 3 \times 3 \times 52 \times 100 \times 32/8 = 187200$ bytes for a 20MHz WiFi channel with 3TX/3RX, 52 subcarriers, and 100 CSI samples with each value represented by 32 bits. Raw CSIs can be compressed by dimension reduction techniques such as Principal/Independent Component Analysis (PCA/ICA), Singular Value Decomposition (SVD), etc., or metrics such as self/cross correlation, Euclidean distance, distribution function, etc. Signal compression can also remove redundant and unrelated information from raw CSI measurements in different domains.

PCA and ICA are widely used for feature extraction and blind signal separation. PCA uses an orthogonal transformation to convert a matrix to a set of principal components. The input is assumed to be a set of possibly correlated variables and the principal components are a set of linearly uncorrelated variables. PCA can be done by SVD or eigenvalue decomposition of the covariance or correlation matrix of the input. ICA assumes that the input signal is a mix of non-Gaussian components that are statistically independent. It maximizes the statistical independence by minimizing mutual information or maximizing non-Gaussianity, i.e., Kurtosis. Many PCA/ICA components can be discarded. For a time series of CSI matrices, redundant measurements can be removed if adjacent samples are highly correlated.

Signal Composition. Some WiFi sensing applications need CSIs from multiple devices, carrier frequency bands, data packets, etc. For example, SpotFi [75] requires CSIs from multiple WiFi devices and multiple data packets to accurately estimate AoAs and ToFs for decimeter-level localization. Chronos [150] requires multiple frequency bands for decimeter-level localization using a single WiFi AP. WiFi sensing algorithms using signal composition are presented in §2.5.1.

2.5 Algorithms of WiFi Sensing

This section presents modeling-based and learning-based algorithms for WiFi sensing. A brief summary and some examples of WiFi sensing algorithms are shown in Table 2.5.

Table 2.5: Summary of WiFi sensing algorithms.

<p>Model: $Y = f(X)$, X: CSI measurements, Y: detection, recognition, or estimation results Algorithm: to find the mapping function $f(\cdot)$ to detect, recognize, or estimate Y given X</p>	
Algorithm Type	Examples
<p>Modeling-based: (1) modeling X by theoretical models based on physical theories or statistical models based on empirical measurements; (2) inferring $f(\cdot)$ by the model of X; (3) predicting Y by the modeled function $f(\cdot)$ and measurements of X, sometimes assisted by optimization algorithms.</p>	<p>Theoretical Models: Fresnel Zone Model, Angle of Arrival/Departure, Time of Flight, Amplitude Attenuation, Phase Shift, Doppler Spread, Power Delay Profile, Multi-Path Fading, Radio Propagation: Reflection, Refraction, Diffraction, Absorption, Polarization, Scattering; Statistical Models: Rician Fading, Power Spectral Density, Coherence Time/Frequency, Self/Cross Correlation; Algorithms: MUSIC, Thresholding, Peak/Valley Detection, Minimization/Maximization</p>
<p>Learning-based: (1) Training: learning $f(\cdot)$ by training samples of X' and Y'; (2) Inference: predicting Y by the learned function $f(\cdot)$ and measurements of X.</p>	<p>Learning Algorithms: Decision Tree, Naive Bayes, Dynamic Time Wrapping, k Nearest Neighbor, Support Vector Machine, Self-Organizing Map, Hidden Markov Models, Convolutional/Recurrent Neural Network, Long Short-Term Memory</p>
<p>Hybrid: (1) modeling the problem by $Y = f(g(X))$; (2) getting $f(\cdot)$ and $g(\cdot)$ by modeling-based or learning-based algorithms; (3) predicting Y by the modeled or learned function $f(g(\cdot))$ and measurements of X.</p>	<p>$g(\cdot)$ by modeling-based algorithms $\rightarrow f(\cdot)$ by learning-based algorithms: e.g., (1) extracting mobility data by Doppler Spread \rightarrow recognizing gestures by k Nearest Neighbor [118]; e.g., (2) estimating position and orientation features by Channel Frequency Response \rightarrow recognizing gestures by k Nearest Neighbor [154]</p>

2.5.1 Modeling-Based Algorithms

Modeling-based algorithms are based on physical theories like the Fresnel Zone model, or statistical models like the Rician fading model.

Theoretical Models. As shown in equation (2.1) in §2.3, CSI is a matrix of complex values representing the CFR of multi-path MIMO channels. CSI amplitude attenuation and phase shift are impacted by the distance between the transmitter and receiver and the multi-path effects including radio reflection, refraction, diffraction, absorption, polarization, and scattering. The amplitude attenuation of Free Space Propagation is

$$P_r/P_t = D_t D_r (\lambda/4\pi d)^2, \quad d \gg \lambda, \quad (2.5)$$

where D_t and D_r are the antenna directivity of the transmitter and receiver, respectively, λ is the carrier wavelength, and d is the distance between the transmitter and receiver. It models wireless signals traveling through free space by the LoS path. In real-world scenarios, there are other objects and humans. According to equation (2.1), the phase shift is impacted by the time delay of each path. Phase shift is also impacted by the Doppler effect when either the transmitter or receiver moves with a speed lower than the velocity of radio waves in the medium. The observed frequency is $f = f_0(c + v_r)/(c + v_t)$, where v_r and v_t are the velocity of the receiver and transmitter, respectively, with respect to the medium, c is the velocity of radio waves, and f_0 is the original carrier frequency. Doppler phase shift is an effective model for motion detection and speed estimation.

CSI amplitude and phase are impacted by radio waves from multiple paths rather than a single path. The Fresnel Zone model divides the space between and around the transmitter and receiver into concentric prolate ellipsoidal regions, or Fresnel zones. The radius of the n -th Fresnel Zone is calculated as shown in Fig. 2.6. It shows how radio signals propagate and deflect off objects within the Fresnel zone regions. The deflected signals travel through multiple paths to the receiver. Depending on the path length and the resulting amplitude attenuation and phase shift, the deflected signals lead to constructive

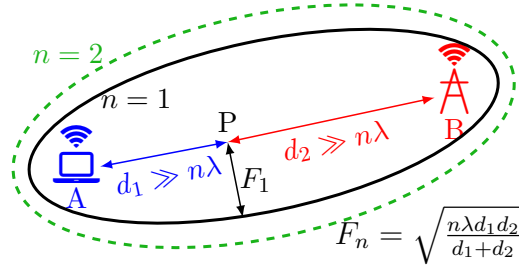


Figure 2.6: Fresnel Zone Model. F_1 is the radius of the first Fresnel zone ($n = 1$) at point P.

or destructive effect at the receiver.

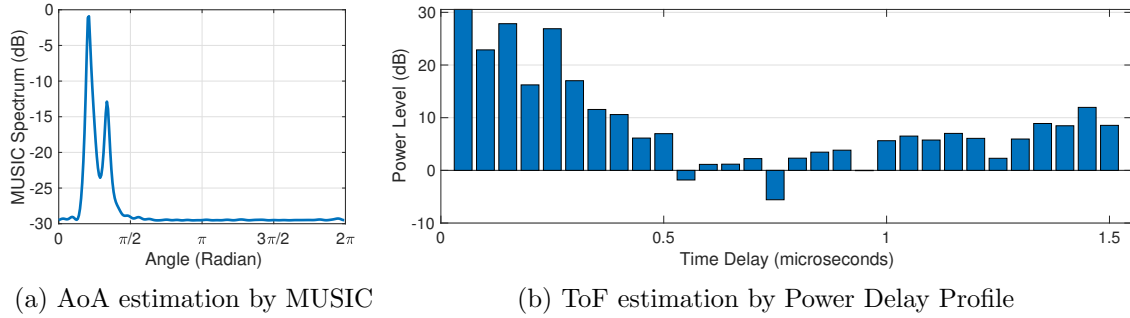


Figure 2.7: Estimation of Angle-of-Arrival and Time-of-Flight by CSI.

AoAs and ToFs are two popular models for CSI-based tracking and localization. They characterize the amplitude attenuation and phase shift of multi-path channels in terms of directions and distances. AoAs and ToFs are estimated by the phase shift or time delay from CSI measurements of an antenna array. The Multiple Signal Classification (MUSIC) algorithm is widely used for estimating AoAs. It computes the Eigen value decomposition of the covariance matrix from CSI [75]. AoAs are calculated based on the steering vectors orthogonal to the Eigen vectors. Fig. 2.7a shows an example of MUSIC spectrum of different AoAs. ToFs can be estimated by Power Delay Profile (PDP) which represents the signal strength of multiple paths with different time delays. PDP is calculated by the Inverse Fast Fourier Transform (IFFT) of CSI. The corresponding PDP of CSI $H(f)$ is $h(t) = \sum_{n=1}^N \alpha_n \delta(t - \tau_n)$, where N is the number of paths, α_n and τ_n are the attenuation

and delay of the n -th path, respectively, and $\delta(\cdot)$ is the impulse function. The norm of $h(t)$ is the signal strength of each path along which the signal arrives at the receiver with time delay t , as shown in Fig. 2.7b.

Statistical Models. Statistical models rely on empirical measurements or probability functions to characterize wireless channels. Rician fading is a stochastic model used by some WiFi sensing applications. It is a simple model for multi-path channels with a dominant path that is stronger than others. The received signal amplitude of a Rician fading channel follows a Rice distribution with $\nu^2 = K\Omega/(1 + K)$ and $\sigma^2 = 2\Omega/(1 + K)$, where K is the ratio between the power in the direct path and the power in the other scattered paths, and Ω is the total power, i.e., $\Omega = \nu^2 + 2\sigma^2$. CSI similarity is a widely used metric for motion-related WiFi sensing applications. It is calculated by the cross correlation of two CSI matrices [47]. Empirical measurements show that CSI similarity is a good indicator of whether the WiFi device and surrounding objects are static or moving [47]. Coherence time and coherence bandwidth, which represent the time duration or bandwidth during which the CIR is coherent, can also be used to detect the mobility status of WiFi devices.

Algorithms for Theoretical and Statistical Models. Threshold-based methods, peak/valley detection, and clustering are widely used modeling-based algorithms for WiFi sensing. Threshold-based methods are simple and effective for amplitude attenuation, cross correlation and distance metrics, especially for detection applications. As shown in Fig. 2.5, RSS and CSI amplitude are in different ranges when the user is making gestures and when the user is static. Different CSI similarity thresholds can also be used to determine the mobility status: if CSI similarity is less than 0.9, the WiFi device is moving; if it is no less than 0.9 but less than 0.99, it is environmental mobility; otherwise, it is static [47]. Threshold-based methods can also be used with other statistical metrics such as variance, Mean Absolute Deviation (MAD), Power Spectral Density (PSD), etc., and distance metrics such as Dynamic Time Wrapping (DTW), Euclidean distance, Earth Mover’s Distance (EMD), etc. Peak/valley detection is widely used for phase shift and

Doppler Spread for WiFi-based respiration and moving speed estimation. In these cases, CSI phases have periodic patterns, which can be detected by peak/valley detection or frequency-domain analysis.

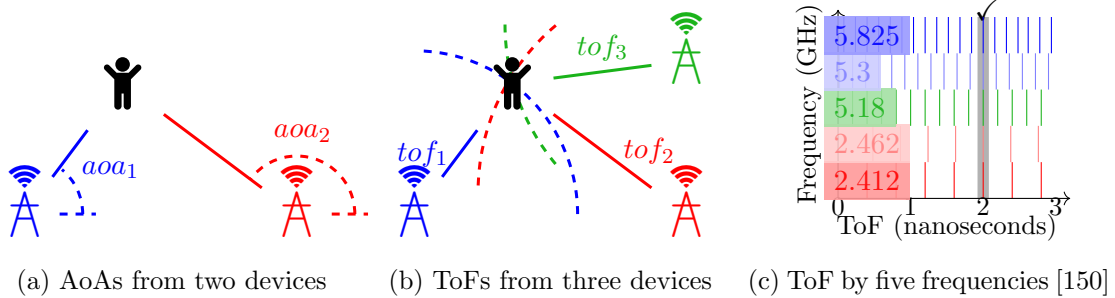


Figure 2.8: Localization by CSIs from multiple WiFi devices and frequency bands. Real-world applications need more than three WiFi devices, assisted by clustering or majority vote, to mitigate noises and estimation errors.

For WiFi sensing using AoAs and ToFs, it usually requires CSI measurements from multiple devices, frequency bands or data packets. SpotFi [75] uses AoAs and ToFs from multiple WiFi APs to localize the target, as shown in Fig. 2.8a and 2.8b. It also measures CSIs by multiple data packets to mitigate the impact of noises and estimation errors. Gaussian mean clustering is used to identify AoAs and ToFs from the same path but different packets. The assumption is that the direct path has the smallest ToF, so a large ToF means a low likelihood to be the direct path. SpotFi selects the path with the highest likelihood as the direct path. Chronos [150] achieves decimeter-level localization with a single WiFi AP. It estimates ToFs from multiple frequency bands such that it does not require multiple WiFi devices. As shown in Fig. 2.8c, a single frequency band gives a set of potential ToFs. The true ToF is identified by the Least Common Multiple (LCM) algorithm.

2.5.2 Learning-Based Algorithms

Binary and multi-class classification applications usually use learning-based algorithms. These algorithms try to learn the mapping function using training samples of CSI mea-

surements and the corresponding ground truth labels.

Shallow Learning Algorithms. Similar to threshold-based methods, Decision Tree (DT) learning tries to find a branching rule to predict the target classes. The difference is that the branching rule of DT is learned from training data instead of hand-crafted. Naive Bayes is another technique for constructing simple and lightweight classifiers based on the Bayes' theorem. A Bayesian network is a probabilistic graphical model that represents the instances and their conditional dependencies by a Directed Acyclic Graph (DAG). Another widely used statistical algorithm is Hidden Markov Model (HMM) which can be regarded as the simplest dynamic Bayesian network. HMM represents the classification problem as a Markov process wherein the true states are hidden.

Instance-based learning algorithms, such as k Nearest Neighbor (kNN), Support Vector Machine (SVM), and Self-Organizing Map (SOM), are widely used for detection and recognition applications. These algorithms compute the distance between each testing sample and every training samples. For kNN, the testing sample is classified by the majority vote of the ground truth labels of its k nearest neighbors. SVM separates data points by a set of hyperplanes in a high dimensional space to maximize the functional margin, i.e., the distance to the nearest training data points of any class. SOM represents training samples in a low dimensional space. It is a type of neural networks using competitive learning instead of backpropagation with gradient descent as the optimization algorithm. A distance metric, such as Euclidean and Hamming distance, is needed for instance-based learning algorithms. Dynamic Time Wrapping (DTW) and data interpolation are widely used when the input is a time series of CSIs with different time durations or number of samples.

The input for shallow learning algorithms could be raw CSIs, pre-processed CSIs, or feature vectors. Since raw CSIs are usually too large and noisy, they rarely serve as the input. Pre-processed CSIs could be the filtered components of CSIs after signal transform techniques such as FFT, STFT, DWT, etc. The output of thresholding and subcarrier selection could also be the input of learning algorithms. Pre-processing helps remove

noises and reduce the input size. Sometimes pre-processed CSIs are still too large and noisy for shallow learning algorithms. Feature engineering helps extract meaningful and compressed information, e.g., domain knowledge, from raw or pre-processed CSIs. It is widely used for shallow learning algorithms such as kNN and SVM. Statistical metrics are commonly used features, and dimension reduction techniques such as PCA, ICA, and SVD can also be used to extract feature vectors. Feature extraction and selection usually have a great impact on the performance of shallow learning algorithms.

Deep Learning Algorithms. For shallow learning algorithms, it is hard to extract and select the right features effectively and efficiently. Deep Neural Networks (DNN) address this problem by learning features automatically. DNNs require very little or none signal processing, feature engineering, and parameter tuning. DNNs are very powerful for multi-class classification applications. A DNN is organized into multiple layers. The output of the i -th layer is represented by

$$\mathbf{y}^{(i)} = g^{(i)} \left(\mathbf{W}^{(i)} \mathbf{x}^{(i)} + \mathbf{b}^{(i)} \right), \quad (2.6)$$

where $\mathbf{x}^{(i)}$ is the input, $\mathbf{W}^{(i)}$ is the weight matrix, $\mathbf{b}^{(i)}$ is the bias vector, and $g^{(i)}$ is the activation function [40]. The output of the previous layer is the input of the current layer, i.e., $\mathbf{x}^{(i)} = \mathbf{y}^{(i-1)}$. The first layer $\mathbf{x}^{(1)}$ is the original input, i.e., raw or pre-processed CSI measurements. The last layer $\mathbf{y}^{(n)}$ is the final output, i.e., binary or multi-class labels. DNNs learn the weights \mathbf{W} and biases \mathbf{b} , using an optimization algorithm, to minimize the cost function. For example, Stochastic Gradient Descent with Momentum (SGDM) is a widely used optimization algorithm that takes small steps in the direction of the negative gradient of the loss function. To prevent overfitting, L2 regularization is usually used to add a regularization term for the weights to the loss function.

A Convolutional Neural Network (CNN) is a DNN with at least one of its layers involving convolution operations. CNNs are effective for learning local features. CNNs are relatively fast to run during training and inference due to shared kernels. CNNs are

proven to have very good performance and are seen in almost all modern neural network architectures. For a sequence or a temporal series of data samples, it is usually better to use 1D CNNs or Recurrent Neural Networks (RNNs). 1D CNNs use one dimensional instead of two dimensional convolution, so they have low computational cost and good performance for simple classification problems. A major characteristic of CNNs is the lack of memory for a sequence or a time series of data points. A RNN has internal connections by iterating through the time series of input elements. Simple RNNs have the vanishing gradient problem that the network becomes untrainable as new layers added to the network [19]. Long Short-Term Memory (LSTM) is an effective and widely used architecture to address this problem. It saves the state information for later units so it prevents previous states from gradually vanishing during training. RNNs with LSTM are usually the right choice for processing a sequence or a time series of data points where temporal ordering matters. The major drawback of RNNs and LSTM is that they have very high computation cost.

A 3D CSI matrix with $size(H) = N \times M \times K$ is similar to a digital image with spatial resolution of $N \times M$ and K color channels, so WiFi sensing can reuse DNNs that have high performance for computer vision tasks. Besides, CSI data have some unique properties that are different from images and videos. For example, CSI has much smaller spatial resolutions and more frequency channels than images. Another challenge is that CSI is impacted by multi-path effects and interferences from all directions, so it contains a lot of noises and is very sensitive to environmental changes. Therefore, WiFi sensing may need new DNN architectures specifically designed for CSI data.

2.5.3 Hybrid Algorithms

Modeling-based and learning-based algorithms have their own advantages and limitations. For example, one of the major limitations of learning-based algorithms is overfitting, since the training process usually can only find the patterns and information that are present in the training data. Different algorithms have different requirements of signal

Table 2.6: Pros and Cons of modeling-based and learning-based algorithms.

modeling-based algorithms	
Pros	<p>need very little or none training data collection, model training, and ground truth labeling</p> <p>need only simple algorithms, e.g., thresholding, peak/valley detection, clustering, etc.</p> <p>usually have low costs and run fast for both off-line analysis and real-time running</p>
Cons	<p>need efforts for building the suitable models and finding the right model parameters</p> <p>need very accurate measurements and estimations, along with a lot of signal processing</p> <p>usually not reusable, versatile, or scalable for new tasks, scenarios, environments, etc.</p>
Use Cases	<p>Mostly used for estimation applications which require accurate estimations of numerical values. Noise removal is crucial for modeling-based algorithms and estimation applications.</p>
learning-based algorithms	
Pros	<p>need very little or none signal processing</p> <p>evolvable: could improve when there are more training data, especially for deep learning</p> <p>automatic for deep learning: no need of feature engineering or learning parameter tuning</p>
Cons	<p>reusable for deep learning: no need to restart training on new data or pre-trained models</p> <p>versatile for deep learning: can reuse high-accuracy pre-trained models from other tasks</p> <p>need a lot of efforts for training data collection and ground truth labeling</p> <p>need a lot of training data in different settings and easy to overfit to the training data</p> <p>need a lot of resources and time for training, especially for deep learning</p> <p>shallow learning: need feature engineering to find and select the right features</p> <p>instance-based learning algorithms, e.g., kNN, have high costs during the inference stage</p>
Use Cases	<p>Mostly used for recognition applications and need very little or none signal processing.</p>

processing techniques and are suitable for different WiFi sensing applications. Modeling-based algorithms are more suitable for estimation applications, and learning-based algorithms are better choices for recognition applications. For detection applications, either modeling-based or shallow learning algorithms can be the right choice. The pros and cons of *modeling-based and learning-based WiFi sensing algorithms* are listed in Table 2.6.

Hybrid algorithms use both modeling-based and learning-based algorithms to address the limitations of each type of algorithms. In some cases, modeling-based algorithms are used to get coarse-grained information and then learning-based algorithms are used for fine-grained and complex tasks. For example, WiSee [118] first extracts mobility data by Doppler phase shift and then recognizes hand and body gestures by kNN. WiAG [154] first estimates the position and orientation features by CFR and then uses kNN to recognize gestures. In some cases, . For estimation applications, learning-based algorithms can be first used to detect or recognize certain events, and then modeling-based algorithms are used to estimate the quantity values of the target events.

2.6 Applications of WiFi Sensing

This section presents a summary and comparison of different WiFi sensing applications, as shown in Table 2.7. The signal processing techniques, algorithms, and performance results are summarized in Table 2.8, 2.9, and 2.10. For signal processing, NR represents Noise Reduction, ST represents Signal Transform, and SE stands for Signal Extraction. Modeling-based and learning-based algorithms are represented by M and L, respectively. Details of which algorithms require what signal processing techniques and are suitable for which types of WiFi sensing applications are also presented.

2.6.1 Detection Applications

Table 2.8 shows the summary of WiFi-based detection applications, most of which are for human presence detection and human event detection. For event detection, most

Table 2.7: Summary of existing WiFi sensing applications.

Output Type	WiFi Sensing Applications
Detection: binary classification	Human Presence Detection [3, 230, 193, 229, 121, 119, 233, 184, 109, 140, 186, 39]
	Human Event Detection: Fall [157, 110, 52, 214, 216], Motion [42, 87, 38, 81], Walking [202], Posture Change [89, 90], Intrusion [81, 91], Sleeping [89, 90], Keystroke [5], Driving Fatigue [113, 25], Lane Change [183], School Violence [227], Smoking [223, 222], Attack [85, 86, 199, 67], Tamper [7], Abnormal Activity [232]
	Object Detection [188]; LoS/NLoS Detection [185, 231]
Recognition: multi-class classification	Activity Recognition: Daily Activities [6, 23, 27, 30, 33, 43, 158, 165, 166, 173, 170, 176, 189], Shopping [211], Driving [25, 125], Exercising [192], Speaking [155], Acoustic Eavesdropping [177], Head & Mouth Activities [29], Walking [96]
	Gesture Recognition: Body/Head/Arm/Hand/Leg/Finger Gestures [153, 123, 203, 213, 2, 3, 53, 118, 146, 154, 219, 228, 95, 79, 100, 136], Sign Language Recognition [79, 95, 100, 136], Keystroke Recognition [4, 5, 78, 80]
	Human/User Identification [16, 17, 54, 164, 198, 212, 218]; Human/User Authentication [85, 86, 137, 163, 190]
	Object Recognition [234, 239, 183]; Object Event Recognition [108]
Estimation: quantity values of size, length, angle, distance, duration, frequency, counts, etc.	Device-Free Human Localization/Tracking: Position [229, 159, 82, 120, 122, 178, 112, 58], Orientation [154, 207], Motion [68, 72, 187, 207], Walking Direction [187, 202, 215, 96], Step/Gait [164, 202], Hand Drawing [207, 208, 144], Speed [216]
	Device-Based Human Localization/Tracking [75, 150, 196, 208]
	Object Localization/Tracking [92, 178, 183]; Humidity Estimation [220]
	Breathing/Respiration Rate Estimation: Single Person [217, 93, 1, 161, 169, 90], Multiple Persons [161, 169]; Heart Rate Estimation [88, 135, 168]
	Human Counting: Static Humans [24, 191], Moving Humans [225, 117, 9, 156, 44], Human Queue Length [172, 171, 183]; WiFi Imaging [70, 55, 235, 234]

Table 2.8: Summary of WiFi sensing: detection applications.

Reference	Signal Processing	Algorithm	Application	Performance
Wi-Vi [3]	NR: Signal Nulling	M: AoA	Moving Human Detection; Gesture Decoding	Human Detection: 85% to 100% (3 humans); Gesture Decoding: 93.75% (6-7m), 75% (8m), 0 (9m)
Gong-2016 [39]	N/A	M: Rician Fading, Cross-Correlation	Human Detection	False Negative: <5%; False Positive: <4%
Palipana-2016 [109]	SE: Interpolation, Kernel PCA	M: Threshold-Based Detection, Rician Fading	Human Detection	True Positive: 90.6%
PADS [121, 119]	NR: Phase Offset, Hampel Filter	L: One-Class SVM	Human Detection	True Positive Rate: >93%
PeriFi [140]	NR: Phase Offsets (PDD, STO)	M: AoA, ToF, MUSIC; L: One-Class SVM	Human Detection	Accuracy: 96.7%
DeMan [184]	NR: Hampel Filter, Linear Fitting, Least Median Squares; SE: Correlation Matrix	M: Sinusoidal Model, Nelder-Mead Searching	Moving & Stationary Human Detection	Detection Rate: 94%/92% (moving/stationary)
Xiao-2015 [193]	NR: WMA	M: Threshold-Based Detection	Human Detection	N/A
Zhou-2017 [229]	NR: Density-Based Spatial Clustering; SE: PCA	L: SVM Classification & Regression	Human Detection & Localization	Detection Accuracy: >97%, Localization Error: 1.22m/1.39m (lab/meeting room)
Zhou-2014 [230]	SE: Feature Extraction	M: EMD, Fingerprinting, Threshold-Based Detection	Human Detection	Average FPR/FNR: 8%/7% (fingerprinting), ~10% (threshold)
R-TTWD [233]	NR: Hampel & Wavelet Filter; ST: DWT; SE: Feature Extraction, Interpolation, PCA	L: Majority Vote, One-Class SVM	Moving Human Detection	True Positive/True Negative: >99%
WiFall [52]	NR: WMA, LOF	L: kNN, One-Class SVM	Fall Detection	Detection Precision: 87%
FallDeFi [110]	NR: Wavelet Filter; ST: DWT, STFT; SE: PCA, Interpolation, Subcarrier Selection, Thresholding	M: Power Burst Curve; L: One-Class SVM	Fall Detection	Accuracy: 93%/80% (same/different testing environments)

Continued on next page.

Table 2.8 Continued 1. Summary of WiFi sensing: detection applications.

Reference	Signal Processing	Algorithm	Application	Performance
RT-Fall [157]	ST: STFT; SE: BPF, Interpolation, Feature Extraction, Thresholding	M: Amplitude Attenuation, Phase Shift; L: One-Class SVM	Fall Detection	True Positive Rate: 91%, True Negative Rate: 92%
Anti-Fall [214]	SE: Interpolation, LPF, Threshold-Based Sliding Window	M: Amplitude Attenuation, Phase Shift; L: One-Class SVM	Fall Detection	Precision: 89%, False Alarm Rate: 13%
WiSpeed [216]	NR: Median Filter; SE: ℓ_1 Trend Filter, Thresholding	M: Multi-Path Scattering, Statistical Modeling, Peak Detection	Fall Detection & Speed Estimation	Fall Detection: 95%, Mean Error: 4.85%/4.62% (device-free/-based)
MoSense [42]	SE: LPF, Euclidean Distance, Thresholding	M: CFR; L: Binary Classification	Motion Detection	Accuracy: 97.38%/93.33% (LoS/NLoS, 5 activities)
Liu-2017 [87]	NR: Phase Difference; SE: Signal Isolation by Skewness	M: CIR; L: One-Class SVM	Motion Detection	Motion Detection Rate: 90.89%
FRID [38]	N/A	M: CFR, Coefficients of CSI Phase Variation	Motion Detection	Precision: 90%
AR-Alarm [81]	SE: Interpolation, BPF, Duration-Based Filter	M: Phase Difference; L: Binary Classification	Motion & Intrusion Detection	True Positive Rate: 98.1%/97.7%
SEID [91]	SE: Signal Compression by CSI Amplitude Variance	M: CFR; L: HMM	Intrusion Detection	Precision: 98%
WiStep [202]	NR: Long Delay Removal; ST: FFT, IFFT, DWT; SE: Butterworth BPF, PCA, Subcarrier Selection	M: Multi-Path Fading, CIR, Short-Time Energy, Peak Detection, Threshold-Based Detection	Walking Detection & Step Counting	Walking Detection: 96.41%/1.38% (TPR/FPR); Step Counting: 90.2%/87.59% (laboratory/classroom)
Wi-Sleep [89, 90]	NR: Hampel Filter, Wavelet Filter; ST: DWT; SE: Interpolation, Subcarrier Selection by Periodicity & SVD, Multiple TX-RX Pairs	M: CFR	Respiration Rate & Apnea Estimation; Posture Change Detection	Respiration Rate Estimation: 85%; Posture Change Detection: 83.3%; Apnea Estimation: 89.8%
WiKey [4, 5]	NR: LPF, PCA; ST: DWT	L: kNN+DTW	Keystroke Detection & Recognition	Detection: 97.5%; Recognition: 96.4% (37 keys)

Continued on next page.

Table 2.8 Continued 2. Summary of WiFi sensing: detection applications.

Reference	Signal Processing	Algorithm	Application	Performance
LiveTag [32]	NR: Signal Nulling; SE: PCA	M: AoA, MUSIC, SSP, SVD, Maximum Likelihood	Touch Detection	Missed Detection Rate: <3% to 28% (LoS), <3% to 14% (NLoS)
Bagci-2015 [7]	NR: MA; SE: Euclidean/ Mahalanobis Distance, EMD, Thresholding	M: Received Signal Strength	Tamper Detection	True Positive Rate: 53%
Liu-2018 [85, 86]	NR: Temporal Bias, De-Correlation Filter, Frequency/Temporal Smoothing; SE: Thresholding, k Means	M: Coherence Time; L: One-Class SVM	Attack Detection, User Au- thentication	Average Attack Detection Ratio: 92%; Authentication Accuracy: 91% (static), 70.6% to 93.6% (mobile)
CSITE [67]	SE: Merging Adjacent Samples	M: Euclidean Distance, Mean Standard Variance, Threshold-Based Detection	Spoofing Attack Detection	False Positive Rate: <4%, False Negative Rate: <4.5%
SecureArray [199]	NR: Random Phase Perturbation	M: AoA, Coherence Time, Threshold-Based Detection	Spoofing Attack Detection	Detection Rate: 100%, False Alarm Rate: 0.6%
WiFind [113]	NR: Hampel Filter, LOF, MA; SE: PCA	L: One-Class SVM	Driver Fatigue Detection	Detection Rate: 82.1%
WiTraffic [183]	NR: Butterworth LPF	L: Threshold-Based Detection, SVM, EMD	Traffic Monitoring	Lane Detection: 95%; Vehicle Recognition: 96%; Speed Error: 5mph
Smokey [222, 223]	NR: Hampel Filter; SE: Interpolation, Antenna Selection, Thresholding	M: Temporal/Frequency Correlation, Peak Detection	Smoking Detection	True Positive Rate: 92.8%, False Alarm Rate: 2.3%
Wi-Dog [227]	ST: DHT, STFT; SE: Antenna/Subcarrier Selection, PCA, Butterworth BPF	M: Doppler Shift, Wavelet Entropy, Median Filter, Thresholding; L: One-Class SVM	School Violence Detection	TPR: 85%/94%, FPR: 11%/10% (classroom/corri- dor)
MAIS [30]	ST: Linear Transform; SE: LPF, Outlier Filter, Thresholding, Eigen Values	L: kNN	Human Counting, Activity Detection & Recognition	Anomaly Detection: 98.04%, Human Counting: 97.21%, Activity Recognition: 93.12%

Continued on next page.

Table 2.8 Continued 3. Summary of WiFi sensing: detection applications.

Reference	Signal Processing	Algorithm	Application	Performance
NotiFi [232]	SE: PCA	L: Nonparametric Bayesian Model, Dynamic Hierarchical Dirichlet Process	Abnormal Activity Detection	Average Accuracy: 89.2%/85.6%/75.3% (LoS/NLoS/through-wall)
PhaseU [185]	NR: Linear Fitting; SE: Thresholding, Antenna Selection	M: Multi-Path Reflections, Diffractions and Refractions	LoS/NLoS Detection	Detection Rate: >94%/80% (static/mobile)
LiFi [231]	NR: CFO; ST: IFFT; SE: Normalization, Thresholding	M: CIR, Rician Fading, PDP, Skewness	LoS/NLoS Detection	Accuracy: 90.4%; False Alarm Rate: 9.34%
Wi-Metal [188]	NR: Interference Nulling by Phase Difference	M: Radio Reflection; L: k Means, Euclidean Distance	Metal Detection	Accuracy: 90%; False Alarm Rate: 10%

are on motion activities, e.g., fall detection. Modeling-based algorithms, e.g., threshold-based detection, and simple learning-based algorithms, e.g., one-class SVM, are widely used. Among the 11 papers on WiFi-based human detection, 5 papers use SVM and 3 papers use threshold-based detection. For the remaining 31 papers, 9 of them use one-class SVM. Theoretical and statistical models are usually sensitive to noises and outliers. Noise reduction is usually needed for modeling-based algorithms. The Hampel filter, wavelet filter, LOF are popular choices. Detection problems are relatively simple to solve and sometimes have no clear borderline between signal extraction and the classifier. After signal extraction such as LPFs and thresholding, the detection results can be directly derived without further detection or classification algorithms. Several papers use PCA to filter out redundant information. Since binary classification problems usually do not need extensive input data, detection applications usually do not need signal compression or feature extraction. Computation overhead is not a major issue for detection applications due to low input data volume and low complexity for the detection algorithms.

2.6.2 Recognition Applications

Table 2.9 shows the summary of WiFi sensing for multi-class classification tasks. Most of the recognition applications are on activity recognition, gesture recognition, and hu-

Table 2.9: Summary of WiFi sensing: recognition applications.

Reference	Signal Processing	Algorithm	Application	Performance
Wi-Chase [6]	SE: LPF, Modulation Filter	M: Path Loss, PDP; L: kNN, SVM	Activity Recognition	Recognition Accuracy: 97% (3 activities)
WIBECAM [23]	N/A	M: PDP, Autoregressive Model, PSD	Activity Recognition	Recognition Accuracy: 73% to 100% (4 activities)
BodyScan [27]	ST: FFT; SE: Butterworth LPF, PCA, Thresholding	M: PSD, Statistical Distribution; L: SVM	Activity Recognition, Breathing Monitoring	Activity Recognition Accuracy: 72.3% (5 activities), Breathing Rate Accuracy: 97.4%
MAIS [30]	ST: Linear Transform; SE: LPF, Outlier Filter, Thresholding, Eigen Values	L: kNN	Human Counting, Activity Detection & Recognition	Anomaly Detection: 98.04%, Human Counting: 97.21%, Activity Recognition: 93.12%
DFLAR [33]	N/A	L: Sparse Auto-Encoder Neural Network	Activity Recognition	Recognition Accuracy: 90% (8 activities)
HuAc [43]	NR: Outlier Filter, WMA; SE: LPF, Thresholding, k Means	L: SVM	Activity Recognition	Recognition Accuracy: 93% (16 activities)
EI [66]	NR: Hampel Filter; ST: FFT; SE: Thresholding	L: Correlation, CNN	Activity Recognition	Accuracy: <75% (10 users, 6 activities, 3 rooms)
Wang-2018 [158]	NR: Median Filter, Linear Fitting; ST: FFT; SE: LPF, Feature Extraction	M: Coherence Histogram; L: SOM, Softmax Regression	Activity Recognition	Recognition Accuracy: >85%
CARM [165, 166]	NR: CFO; ST: DWT; SE: Thresholding, PCA, Feature Extraction	L: HMM	Activity Recognition	Recognition Accuracy: >96% (8 activities)
Wang-2015 [173]	NR: Gaussian Filter, LOF; SE: k Means, Feature Selection	M: Free Space Propagation Model; L: DTW, SVM	Activity Recognition & Fall Detection	Activity Recognition: 80% (13 activities); Fall Detection: 95.2%
E-eyes [170]	NR: LPF, MCS Filter; SE: EMD, Thresholding, Clustering, Multiple Links	L: Multi-Dimensional DTW, Pattern Matching	Activity Recognition	Average Recognition Accuracy: 90%/95% (single device/multiple devices, 13 activities)
Wei-2015 [176]	NR: Exponential Smoothing	L: Sparse Representation	Activity Recognition	Recognition Accuracy: 90% (8 activities)

Continued on next page.

Table 2.9 Continued 1. Summary of WiFi sensing: recognition applications.

Reference	Signal Processing	Algorithm	Application	Performance
ARM [189]	NR: CFO, Wavelet Filter; ST: DWT	L: DTW, HMM	Activity Recognition	Average Accuracy: >75% (6 activities)
Zeng-2015 [211]	SE: BPF, Feature Extraction, Multiple APs	M: CFR; L: DT, Simple Logistic Regression	Shopper Activity Recognition	Average Accuracy: 89.6%/94.75 (entrance/in store, 4 activities)
WiDriver [25]	SE: Signal Compression by Back Propagation Neural Network	M: Fresnel Zone Model; L: Finite Automata	Driver Activity Recognition	Recognition Accuracy: 96.8% (11 postures), 90.76% (7 activities)
HeadScan [29]	SE: Butterworth LPF, PCA	L: Sparse Representation, ℓ^1 Minimization	Head & Mouth Activity Recognition	Recognition Accuracy: 86.3% (5 activities)
SEARE [192]	NR: LPF, Median Filter, PCA Filter; ST: FFT; SE: Thresholding	L: First-Order Difference, DTW	Exercise Activity Recognition	Average Accuracy: 97.8%/91.2% (LoS/NLoS, 4 activities)
WiSome [203]	NR: LOF, Wavelet Filter; ST: DWT, STFT; SE: Locally Linear Embedding, Multiple TXs	M: Doppler Shift, Thresholding; L: kNN, SVM	Motion Direction Recognition	Average Accuracy: 95.4%/95.9%/95.5% (thresholding/kN-N/SVM)
APsense [213]	SE: Feature Extraction	L: Naive Bayes, DT	Motion Recognition	Average TPR: 74.8% (decision tree), 56.8% (naive bayes)
WiDance [123]	ST: STFT; SE: Antenna Selection, Butterworth BPF, PCA, Thresholding	M: Doppler Shift, Rule-Based Classification	Motion Direction Recognition	Accuracy: 92% (9 motion directions)
Maheshwari-2015 [96]	NR: LPF; SE: Cumulative MSD	L: DT	Gait Rate Classification	Accuracy: <60% (3 speeds), >90% (2 speeds)
WiHear [155]	NR: Butterworth BPF; ST: IFFT, DWT	M: Multi-Path Reflection, PDP; L: DTW, Pattern Matching	Speaking Recognition	Accuracy: 91%/74% (1 person/3 persons, <6 words)
ART [177]	NR: Averaging; SE: BPF	M: Wireless Vibrometry	Acoustic Eavesdropping	Recognition Accuracy: 80% (distance<4m)
WiGest [2]	NR: Wavelet Filter; ST: FFT, DWT; SE: Thresholding	L: Pattern Matching	Gesture Recognition	Recognition Accuracy: 87.5%/96% (1 AP/3 APs, 7 gestures)

Continued on next page.

Table 2.9 Continued 2. Summary of WiFi sensing: recognition applications.

Reference	Signal Processing	Algorithm	Application	Performance
Wi-Vi [3]	NR: Signal Nulling	M: AoA	Moving Human Detection; Gesture Decoding	Moving Human Detection: 85% to 100% (3 humans); Gesture Decoding: 93.75% (6-7m), 75% (8m), 0 (9m)
WiG [53]	NR: Birge-Massart Filter, Wavelet Filter, LOF	L: SVM	Gesture Recognition	Recognition Accuracy: 92% (LoS), 88% (NLoS)
WiSee [118]	NR: CFO; ST: FFT; SE: BPF, Interpolation	M: Doppler Shift; L: Pattern Matching	Gesture Recognition	Average Accuracy: 94% (9 gestures)
WiFinger [146]	NR: Wavelet Filter, Butterworth BPF; ST: IFFT, DWT; SE: PCA, Subcarrier Selection	L: Pattern Matching, Multi-Dimensional DTW	Finger Gesture Recognition	Accuracy: 93% (8 finger gestures)
WiMU [153]	ST: STFT; SE: PCA, Thresholding	M: Pattern Matching, Threshold-Based Detection	Multi-User Gesture Recognition	Accuracy: 95.0%, 94.6%, 93.6%, 92.6%, 90.9% (2, 3, 4, 5, 6 concurrent gestures)
WiAG [154]	NR: Butterworth Filter; ST: DWT; SE: PCA, Thresholding, Extrapolation	M: CFR; L: kNN	Gesture Recognition	Accuracy: 91.4% (6 gestures)
Mudra [219]	NR: MA, Finite Impulse Response Filter; ST: FFT, IFFT; SE: Thresholding	L: DTW	Finger Gesture Recognition	Average Accuracy: 96% (9 finger gestures)
DeNum [228]	SE: BPF Feature Extraction	M: Threshold-Based Sliding Window; L: NN, SVM	Gesture Recognition	Average Accuracy: 94% (10 finger postures)
WiFinger [79]	NR: Hampel Filter, LPF, WMA; ST: DWT	M: CFR, PCA; L: kNN+DTW	Sign Language Recognition	Recognition Accuracy: 90.4% (9 hand postures)
SignFi [95]	NR: STO/SFO, Multiple Linear Regression	L: CNN	Sign Language Recognition	Accuracy: 94.8% (276 signs, 1 user, lab+home), 86.6% (150 signs, 5 users, lab)

Continued on next page.

Table 2.9 Continued 3. Summary of WiFi sensing: recognition applications.

Reference	Signal Processing	Algorithm	Application	Performance
Melgarejo-2014 [100]	NR: LPF; SE: Subcarrier Selection by Similarity	L: kNN+DTW	Sign Language Recognition	Accuracy: 84% (14 signs, car), 92% (25 signs, wheelchair)
WiSign [136]	NR: Median Filter, LPF; ST: FFT; SE: Subcarrier Selection, Multiple RXs	L: SVM, Majority Vote	Sign Language Recognition	Mean Accuracy: 93.8% (5 sign gestures)
WiKey [4, 5]	NR: LPF, PCA; ST: DWT	L: kNN+DTW	Keystroke Detection & Recognition	Detection: 97.5%; Recognition: 96.4% (37 keys)
ClickLeak [78]	ST: DWT; SE: LPF, PCA, Thresholding, k Means	L: kNN+DTW	Keystroke Recognition	Recognition Accuracy: 83% (10 keys)
WindTalker [80]	SE: LPF, PCA, Thresholding; ST: DWT	M: CFR; L: DTW	Keystroke Recognition	Accuracy: 81.8%/73.2%/64% (Xiaomi/Nexus/Samsung, 10 numbers)
Rapid [16]	NR: CFO, Hampel Filter, MA; ST: FFT, STFT; SE: Butterworth LPF, Thresholding	M: CFR; L: SVM	Human Identification	Identification Accuracy: 82% to 92% (2 to 6 people)
NiFi [17]	NR: Butterworth LPF, Median Filter; SE: Sequence Similarity	L: Pattern Matching, HMM	User Identification	True Positive Rate: 90.83% (4 devices)
WFID [54]	NR: Threshold-Based Filter; SE: PCA	M: Doppler Shift, Radio Scattering; L: SVM	Human Identification	Identification Accuracy: 93.1% (6 subjects), 91.9% (9 subjects)
WifiU [164]	NR: CFO; ST: STFT; SE: Gaussian LPF, Thresholding, PCA	L: SVM, One-vs-All Classifiers	Human Recognition	Recognition Accuracy: 79.28%/89.52%/93.05% (top-1/-2/-3, 50 subjects)
FreeSense [198]	ST: DWT; SE: PCA, Butterworth LPF, Feature Extraction, Thresholding	L: Mean Absolute Deviation, DTW, kNN	Human Identification	94.5% to 88.9% (2 to 6 users)
WiWho [212]	NR: Distant Multi-path Removal; ST: FFT; SE: Feature Extraction	M: CFR, CIR, Peak-Valley Detection; L: DTW, DT	Human Identification	92% to 80% (2 to 6 users)

Continued on next page.

Table 2.9 Continued 4. Summary of WiFi sensing: recognition applications.

Reference	Signal Processing	Algorithm	Application	Performance
WiFi-ID [218]	NR: Silence Removal; SE: Feature Extraction	L: Sparse Representation	Human Identification	N/A
Liu-2018 [85, 86]	NR: Temporal Bias, De-correlation Filter, Frequency/Temporal Smoothing; SE: k Means, Thresholding	M: Coherence Time; L: SVM	Attack Detection, User Authentication	Average Attack Detection Ratio: 92%; Authentication Accuracy: 91% (static), 70.6% to 93.6% (mobile)
Shi-2017 [137]	ST: FFT; SE: BPF, Subcarrier Selection	L: SVM, Neural Network with Auto-Encoder	User Authentication	Accuracy: 94%/91% (walking/stationary, 11 subjects)
PriLA [163]	N/A	M: CFO, DTW	User Authentication	Average Accuracy: 93.2%
TDS [190]	SE: Feature Extraction by SVD	L: Pearson Correlation, Max-Weighted Bipartite Matching	User Authentication	Error Rate: <7% (authenticate distance=5cm)
WiTraffic [183]	NR: Butterworth LPF	L: Threshold-Based Detection, SVM, EMD	Traffic Monitoring	Lane Detection: 95%; Vehicle Recognition: 96%; Speed Error: 5mph
Ulysses [234]	NR: Majority Vote	M: Specular Reflection, AoA, AoD, Threshold-Based Detection	Object Recognition & WiFi Imaging	Top-3 Accuracy: 100% (11 objects); imaging error: \pm 8cm/1 degree (width/orientation)
TagFree [239]	SE: Feature Extraction	M: Spectral Regression Discriminant Analysis, Random Subspace Method, LDA	Object Recognition	Average Accuracy: 96%/75%/57% (1/2/3 objects, same location, 6 objects)
Ohara-2017 [108]	SE: Signal Separation by ICA	M: CNN, RNN, HMM, LSTM	Object Event Recognition	Average Precision: 81.7%, Recall: 92.5%, F-score: 85.8%

man/user identification and authentication. The number of classes of most recognition applications is about 10. Almost all the recognition applications use learning-based algorithms as the classifier. SVM is still one of the most used algorithms as the classifier. Recognition applications use multi-class SVM instead of one-class SVM for detection applications. Another two widely used classifiers are kNN and DTW. DTW is usually used for kNN as the distance metric. Among the 39 papers on activity and gesture recognition, 8 use SVM, 9 use kNN, and 12 use DTW as the classifier. SVM is the classifier of 6 papers among the 12 papers on human/user identification and authentication. There are several recognition applications using HMM or CNN as the classifier. Many recognition applications use hybrid algorithms which usually first extract information using modeling-based algorithms and then recognize the targets using learning-based algorithms.

Learning-based algorithms are usually not so sensitive to noises and outliers as modeling-based algorithms. Many recognition applications use no or very simple noise reduction methods such as averaging and median filter, instead of complex algorithms such as the Hampel filter and LOF. Noise reduction is used for hybrid algorithms wherein modeling-based algorithms could be sensitive to noises. SVM and kNN are instance-based learning algorithm which need to calculate the distance from the testing instance to all the training instances. This could introduce expensive overhead when there are multiple classes and each class instance has many CSI data points. Many recognition applications, especially those using SVM, kNN, and/or DTW as the classifier, usually employ feature extraction, subcarrier selection, or dimension reduction to reduce the input size.

2.6.3 Estimation Applications

The summary of WiFi-based estimation applications is presented in Table 2.10. For estimation applications, most papers are on human/object localization and tracking. There are also many papers on the estimation of breathing rate, heart rate, and human counts. There are four papers using WiFi for wireless imaging. Different from detection/recognition applications aiming for binary/multi-class classification problems, estimation appli-

Table 2.10: Summary of WiFi sensing: estimation applications.

Reference	Signal Processing	Algorithm	Application	Performance
LiFS [159]	SE: Thresholding	M: Fresnel Zone Model, DTW, Gradient Descent, Genetic Algorithm	Device-Free Human Localization	Median Accuracy: 0.5m (LoS), 1.1m (NLoS)
Zhou-2017 [229]	NR: Density-Based Spatial Clustering; SE: PCA	L: SVM Classification/Regression	Presence Detection & Localization	Presence Accuracy: >97%, Localization Error: 1.22m/ 1.39m (lab/meeting room)
IndoTrack [82]	NR: Phase Offset Removal; SE: Isolating Direct Path Signals, Thresholding	M: Doppler Shift, AoA, MUSIC	Human Tracking	Median Tracking Error: 35cm
Widar [122, 120]	ST: STFT; SE: Butterworth BPF, PCA	M: Doppler Shift, Path Length Change Rate, Searching with Least Fitting Error	Human Tracking	Median Location Error: 25cm/38cm (with/without initial positions); Median Velocity Error: 13%
WiDeo [68]	NR: Thresholding, Full Duplex Interference Nulling	M: AoA, ToF; Kalman Filter, Compressive Sensing	Motion Tracking	Median Error: <7cm for 5 humans
QGesture [207]	NR: CFO, SFO, PBD, MA; ST: DHT; SE: Interpolation, Linear Regression, PCA, Thresholding	M: Multi-Path Propagation, CIR	1D & 2D Motion Tracking	Average Distance Accuracy: 3cm/3.7cm (1D/2D); Average Direction Error: 5%/15 degrees (1D/2D)
WiDir [187]	NR: Cross-Correlation Denoising, Polynomial Smoothing Filter; ST: FFT; SE: Thresholding	M: Fresnel Zone Model, Phase Shift, Radio Reflection/Diffraction	Moving Direction Estimation	Median Error: <10 degrees
WiStep [202]	NR: Long Delay Removal; ST: FFT, IFFT, DWT; SE: Butterworth BPF, PCA, Subcarrier Selection	M: CIR, Short-Time Energy, Peak Detection, Threshold-Based Detection	Walking Detection & Step Counting	Walking Detection: 96.41%/1.38% (TPR/FPR); Step Counting: 90.2% (lab), 87.59% (classroom)
Zhang-2017 [215]	SE: Multiple Carrier Frequencies	M: Fresnel Zone Model	Walking Direction Estimation	Median Error: 10 degrees
WiDraw [144]	SE: Thresholding, Multiple TXs, Transmitter Selection by CSI Correlation	M: AoA, MUSIC	Hand Tracking	Hand Tracking Error: <5cm; Handwriting Accuracy: 91%

Continued on next page.

Table 2.10 Continued 1. Summary of WiFi sensing: estimation applications.

Reference	Signal Processing	Algorithm	Application	Performance
WiSpeed [216]	NR: Median Filter; SE: ℓ_1 Trend Filter, Thresholding	M: Multi-Path Scattering, Statistical Modeling, Peak Detection	Speed Estimation & Fall Detection	Mean Error: 4.85%/4.62% (device-free/-based), Fall Detection: 95%
SpotFi [75]	NR: Sampling Time Offset; SE: Signal Isolation, Multiple Packets and Transmitters	M: AoA, ToF, MUSIC, CSI Smoothing, Gaussian Mean Clustering	Device- Based Localiza- tion	Median Localization Accuracy: 40cm
Chronos [150]	NR: Phase Offsets, PDD; SE: Multi-Path Separation, Multiple Frequency Bands	M: PDP, ToF, Least Common Multiple, Quadratic Optimization	Device- Based Localiza- tion	Median Distance Error: 14.1cm/20.7cm (LoS/NLoS)
Splicer [196]	ST: IFFT; SE: Multiple Carrier Frequencies	M: PDP, MUSIC	Device- Based Localiza- tion	Median Error: 0.95m
AAMouse [208]	NR: Maximal Ratio Combining; ST: STFT; SE: Kalman Filter	M: Doppler Shift	Device- Based Tracking	Median Error: 1.4cm (2 speakers), 2.5cm (1 speaker+WiFi)
BikeLoc [92]	SE: Multiple TXs	M: AoA	Bike Lo- calization	Median Error: 45cm (2 APs); 18.1cm (8 APs)
mTrack [178]	SE: Direct Component Filter, Thresholding	M: Phase Shift, Radio Reflection/Diffusion	Object Tracking	Median Tracking Error: 6.5mm
WiTraffic [183]	NR: Butterworth LPF	L: Threshold-Based Detection, SVM, EMD	Traffic Monitoring	Lane Detection: 95%; Vehicle Recognition: 96%; Speed Error: 5mph
WiHumidity [220]	N/A	M: Radio Absorption, Amplitude Attenuation; L: SVM	Humidity Estimation	Average Accuracy: 79%
UbiBreathe [1]	NR: Local Mean Removal, α -Trimmed Mean Filter; ST: FFT, DWT; SE: BPF, Thresholding	M: dominant periodic component due to inhaling and exhaling	Breathing Rate & Apnea Estimation	breath rate error: 1bpm; breath apnea accuracy: 96%
BodyScan [27]	ST: FFT; SE: Butterworth LPF, PCA, Thresholding	M: PSD, Statistical Distribution; L: SVM	Activity Recogni- tion, Breathing Monitoring	Recognition Accuracy: 72.3% (5 activities), Breathing Rate Accuracy: 97.4%

Continued on next page.

Table 2.10 Continued 2. Summary of WiFi sensing: estimation applications.

Reference	Signal Processing	Algorithm	Application	Performance
Liu-2015 [88]	NR: Hampel Filter, MA; ST: FFT; SE: BPF, Subcarrier Selection by CSI Amplitude Variance, Thresholding	M: Radio Scattering, Fading, and PDP, k Means by PSD	Breathing & Heart Rate Estimation	Breathing Rate Error: <1.1bpm (1 person), <1.2bpm (2 persons); Heart Rate Error: <5bpm (1 person)
Wi-Sleep [89, 90]	NR: Hampel Filter, Wavelet Filter; ST: DWT; SE: Interpolation, Subcarrier Selection by Periodicity and SVD, Multiple TX-RX Pairs	M: CFR	Respiration Rate & Apnea Estimation; Posture Change Detection	Respiration Rate Estimation: 85%; Posture Change Detection: 83.3%; Apnea Estimation: 89.8%
Ma-2016 [93]	NR: Hampel Filter, MA	M: Fresnel Zone Model	Respiration Estimation	N/A
WiHealth [135]	NR: Median Filter, LPF; SE: BPF, Polynomial Filter, Thresholding	M: Multi-Path Fading, Small Scale Fading	Breathing & Heart Rate Estimation	Estimation Error: 0.6bpm (breathing rate), 6bpm (heart rate)
Wang-2016 [156]	NR: Hampel Filter, MA; SE: Thresholding, Subcarrier Selection, Signal Separation	M: Fresnel Zone Model, PSD	Breathing Rate Estimation	N/A
TinySense [161]	ST: IFFT; DWT; SE: Thresholding, Mean Filter, Wavelet Filter, Multiple TX-RX Pairs	M: Fresnel Zone Model, ToF	Multi-Person Breathing Estimation	Accuracy: >88% (2 persons)
PhaseBeat [168]	NR: Hampel Filter, PBD, SFO, CFO; ST: FFT, DWT; SE: Subcarrier Selection, Thresholding	M: CFR, Phase Difference, MUSIC	Breathing & Heart Rate Estimation	Estimation Error: <0.85bpm (breathing rate), <10bpm (heart rate)
TensorBeat [169]	NR: Hampel Filter, PBD, SFO, CFO; SE: Thresholding	M: Phase Difference; L: Canonical Polyadic Decomposition, DTW, Dynamic Programming	Multi-Person Breathing Estimation	Estimation Error: <0.9bpm/1.9bpm (1 person/5 persons)
Zhang-2018 [217]	N/A	M: Fresnel Zone Model, Radio Diffraction	Respiration Estimation	Estimation Accuracy: 61.5% to 98.8%
Domenico-2016 [24]	SE: Euclidean Distance	L: Linear Discriminant Classifier	Human Counting	Recognition Accuracy: 52% to 74% (7 persons)

Continued on next page.

Table 2.10 Continued 3. Summary of WiFi sensing: estimation applications.

Reference	Signal Processing	Algorithm	Application	Performance
MAIS [30]	ST: Linear Transform; SE: LPF, Outlier Filter, Thresholding, Eigen Values	L: kNN	Human Counting, Activity Detection & Recognition	Anomaly Detection: 98.04%, Human Counting: 97.21%, Activity Recognition: 93.12%
FCC [191]	SE: Multiple RXs	M: Rician Fading, Grey Verhulst Model, Percentage of Zero Elements	Human Counting	Error: <3/5 persons (indoor/outdoor, 15 total persons)
Mohammad-moradi-2017 [102]	SE: Signal Compression by Averaging	M: Threshold-Based Hierarchy, Signal to Noise Ratio	Room Occupancy Estimation	Accuracy: 89% (up to 3 persons)
Guo-2017 [44]	NR: ; ST: FFT; SE: LPF, Subcarrier Selection	M: Phase Difference, CSI Variance, EMD, Total Harmonic Distortion	Human Dynamics Monitoring	Accuracy: >90% (number, density, speed, and direction)
Wang-2014 [172, 171]	NR: Dynamic Exponential Smoothing Filter; SE: Interpolation, Thresholding	L: Linear Regression, Feature-Driven Estimation, Bayesian Network, Directed Acyclic Graph	Human Queue Estimation	Estimation Error: <10 seconds (up to 180 seconds queue length)
Wision [55]	ST: FFT; SE: Interference Nulling, Multiple TXs	M: AoA, Diffuse/Specular Radio Reflections, Diffraction	WiFi Imaging	Median Localization Accuracy: 26cm (static human); 15cm (metallic objects)
Karanam-2017 [70]	N/A	M: Markov Random Field Modeling, Loopy Belief Propagation, Sparse Representation	WiFi Imaging	Distance Error: 1.35% to 3.7%
Ulysses [234]	NR: Majority Vote	M: Specular Reflection, AoA, AoD, Threshold-Based Detection	Object Recognition; WiFi Imaging	Top-3 Accuracy: 100% (11 objects); imaging error: <8cm/1 degree (width/orientation)
Zhu-2015 [235]	SE: Thresholding	M: AoA, Radio Reflection, Absorption & Scattering, Majority Vote	WiFi Imaging	Estimation Error: <4.5cm/1 degree (width/orientation)

cations try to calculate the quantity values of size, length, angle, distance, duration, etc. Almost all the estimation applications use modeling-based algorithms, such as AoA, ToF, Fresnel Zone Model, Doppler Spread, MUSIC, etc. For all the 19 papers on human/object localization and tracking, 5 use AoA, 6 use Doppler/Phase Shift, 3 use Fresnel Zone Model. Among 12 papers on breathing/heart rate estimation, 4 use Fresnel Zone Model. Only 6 papers of estimation applications, including 1 on human localization [229], 1 on vehicle speed estimation [183], and 4 on human counting [24, 30, 172, 171], employ only the learning-based algorithms but no modeling-based algorithms. Since modeling-based algorithms are sensitive to noises, estimation applications usually require many efforts on removing noises, especially phase offsets. Many estimation applications employ signal composition techniques, e.g., multiple WiFi devices, frequency bands and data packets, to improve the estimation accuracy.

2.7 Challenges and Future Trends of WiFi Sensing

Existing WiFi sensing mostly focuses on humans. Future WiFi sensing could be in other domains, such as detecting, recognizing, and estimating the surrounding environments, animals, and objects. This section presents the challenges and future trends for both existing and future WiFi sensing. New opportunities for signal processing techniques and algorithms of WiFi sensing are also presented.

2.7.1 Challenges for WiFi Sensing

Robustness and Generalization. WiFi signals are very sensitive to many different factors such as network settings, environments, objects, humans, geometry and mobility situations, etc. It is crucial and also challenging for WiFi sensing to be robust in different real-world scenarios and settings. For example, the distance between the person and the WiFi transmitter/receiver could be different. The direction and orientation of the person with respect to the WiFi transmitter/receiver could also change. There could be multiple

persons or other moving objects around. The person or other objects could block the direct path between the transmitter and receiver. It is more challenging for WiFi sensing algorithms, both modeling-based and learning-based, to have the generalization ability of properly and automatically adapting to new and previously unseen data. For example, WiFi-based activity recognition should also work when WiFi devices are placed in a new environment at unknown locations/orientations and for new persons whose data are not seen before. Learning-based algorithms also have under-fitting issues when there are not enough training data. To guarantee the robustness and generalization of WiFi sensing, it requires effective and efficient ways to find the right data collection methods, signal processing techniques, theoretical/statistical models, and machine learning algorithms.

Privacy and Security. One of the advantages of WiFi sensing is that it is non-intrusive and non-obtrusive. But this introduces many privacy and security issues. As shown in §2.6, there are already many WiFi sensing applications that can infer both coarse-grained and fine-grained information such as daily activities, gestures, and keystrokes. These information can be easily leaked to malicious hackers and attackers. Moreover, the victim user may be unaware of the information leakage since it is non-obtrusive and WiFi signals can travel through walls. Unlike images and videos, WiFi signals are not limited to lighting conditions, so WiFi sensing is very easy to be used for malicious purposes. This could be in conflict with the purpose of robustness and generalization of WiFi sensing: the former one needs to make it harder to leak information while the latter requires more information to be easily inferred in different scenarios. Therefore, new protocols, policies, architectures, and algorithms are needed for the privacy and security of WiFi sensing.

Coexistence of WiFi Sensing and Networking. WiFi is designed for wireless communications but not for sensing applications. When a WiFi device is used for sensing, it could influence the network performance and also be impacted by network settings. Some WiFi sensing applications require high CSI measurement frequency to get high performance. This could introduce overhead for WiFi networks and result in reduced network performance and efficiency. Moreover, sending unnecessary CSI measurement

packets influences not only the measurement device but also other nearby WiFi devices, since it occupies WiFi resources and influences the scheduling process in time and spectrum domains. On the other hand, WiFi sensing is impacted by WiFi network settings. For example, WiFi transmitters may use beamforming which changes the amplitude and phase of CSI measurements, as shown in equation (2.2). This completely changes CSI patterns and is very hard to process if the beamforming matrix is not available at the receiver.

2.7.2 Future WiFi Sensing Trends

This section presents future WiFi sensing trends for addressing the above-mentioned challenges for both existing and future WiFi sensing, as shown in Fig. 2.9.

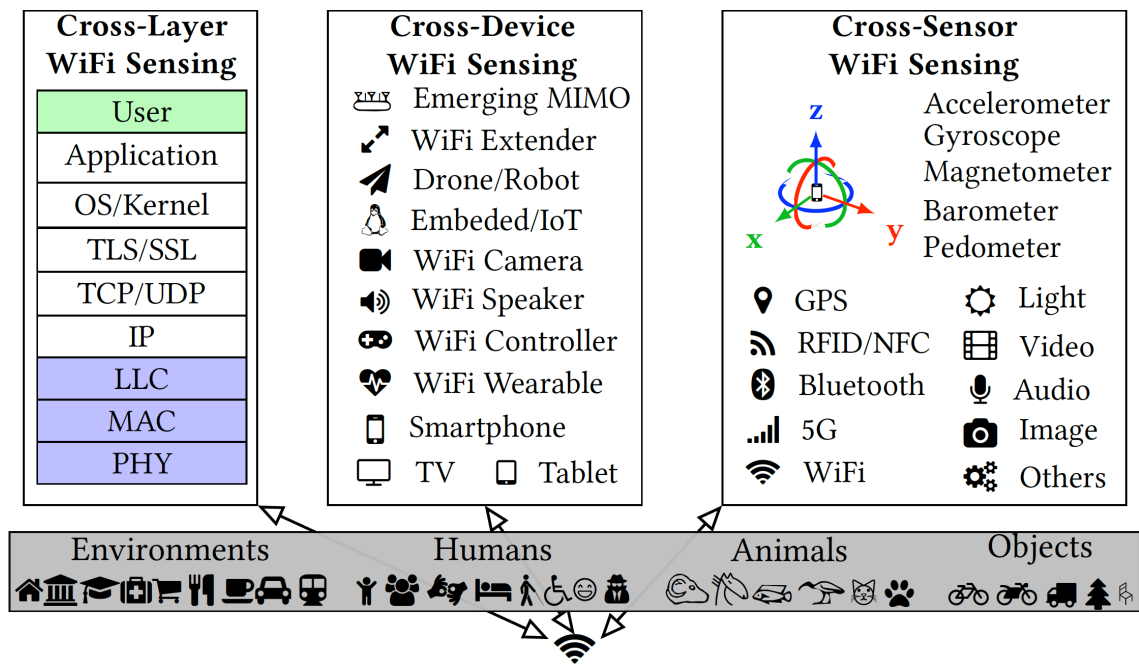


Figure 2.9: Future trends of WiFi sensing. CSI from WiFi can be used to sense the surrounding environments, humans, animals, and objects using cross-layer information, multiple devices, and fusion of different sensors.

Cross-Layer WiFi Sensing. This survey only focuses on WiFi sensing with the physical layer information, i.e., CSI. CSI can be integrated with upper layer information for cross-layer WiFi sensing. This could help develop new sensing applications or enhance

existing WiFi sensing applications. Upper layer WiFi information, such as Medium Access Control (MAC), Transmission Control Protocol (TCP), and Internet Protocol (IP), can also be used for sensing purposes. For example, MAC and IP packet headers from WiFi probing requests can be used to predict smartphone screen on/off [64], human flow [225, 117, 9, 226], urban mobility [20], and social relationship [74, 9]. Combining CSI with MAC and IP layer information could help enhance the capability of WiFi sensing. Cross-layer WiFi sensing provides additional information from other domains, which can improve the robustness and generalization of WiFi sensing. Cross-layer WiFi sensing can also be used for improving security and privacy. There are already many papers on CSI-based user identification/authentication [16, 17, 54, 164, 198, 212, 218, 85, 86, 137, 163, 190] and other security and privacy purposes [8, 80, 199]. These applications can be improved by incorporating CSI with upper layers such as Transport Layer Security (TLS), Secure Sockets Layer (SSL), application layer, and user interface. Upper WiFi layers can also be re-designed to guarantee WiFi sensing is not misused for malicious purposes. Finally, cross-layer WiFi information can help WiFi sensing and networking be aware of each, so it helps address the coexistence of WiFi sensing and networking.

Cross-Device WiFi Sensing. Some WiFi-based localization and tracking applications use CSIs from multiple WiFi devices. Other WiFi sensing applications can also combine multi-device CSIs for higher performance and efficiency. In addition to WiFi APs, many other WiFi-enabled devices, e.g., cameras, speakers, drones, robots, Internet of Things (IoT) devices, etc., can be used. Due to the rapid development and high demand of wireless data, there will be more WiFi devices in different scenarios, such as home, office, school, outdoor, stadium, shopping malls, etc. These WiFi devices have time and location dependence which could provide more information for WiFi sensing. Moreover, CSI measurements can be collected by emerging MIMO technologies such as distributed, cooperative, massive, 3D, and full dimension MIMO [236]. Current WiFi sensing applications only use CSIs measured by traditional MIMO systems. CSIs of emerging MIMO technologies could open new opportunities for WiFi sensing in terms of signal pro-

cessing techniques, channel models, learning algorithms, application types. Platforms for measuring CSIs of these emerging MIMO technologies are also needed for WiFi sensing purposes. Cross-device WiFi sensing provides more information in different domains, e.g., time, space, frequency, user, etc. It also gives cross-correlation and dependence information among multiple devices. The cross-device information is useful for improving the robustness and generalization of WiFi sensing.

Cross-Sensor WiFi Sensing. Some sensing applications use the fusion of CSIs with other signals, such as videos and audios, as the input [102, 16, 65]. CSIs can be combined with other sensor sources, e.g., Bluetooth, 5G, ZigBee, GPS, microphones, image/video cameras, motion sensors, etc., for cross-sensor WiFi sensing. For example, video cameras and CSIs can be combined together for higher performance and less human efforts of training machine learning algorithms. When the light condition is good, video cameras can be used for ground truth labeling for the machine learning algorithms that use CSIs as the input. The CSI-based learning algorithms can be activated when video cameras are not reliable due to poor light conditions. The fusion of video cameras and CSIs can provide a better time coverage than they are used separately. Moreover, the human efforts of data collection, ground truth labeling, and model training can be significantly reduced. There are many pre-trained neural networks that use videos as the input. These video-based neural networks can provide near human-level performance which can be used to automatically label CSI measurements. This could save a lot of time and computation resources for training the machine learning algorithms. The fusion of WiFi and other sensors also helps improve the robustness and generalization of WiFi sensing by integrating information from other domains.

Finally, all these WiFi sensing trends can be integrated to provide multi-domain knowledge. For example, wireless drones and robots have the whole WiFi network stack, multiple cooperative devices, and different sensors. They can combine cross-layer network information, multi-device cooperation, and fusion of different sensors for improving existing WiFi sensing applications and enabling new WiFi sensing opportunities.

2.7.3 Future Opportunities for Signal Processing and Algorithms of WiFi Sensing

Future WiFi sensing trends also bring new opportunities and challenges for signal processing techniques and classification/estimation algorithms. Existing noise reduction techniques mostly focus on removing noises, interferences, and unintended signals for a single device. New noise reduction techniques and hardware designs are needed to deal with noise signals from multiple devices and other domains. Since there are multi-domain signals from upper network layers, multiple devices, and sensor fusions, new signal compression techniques are needed to remove redundant and unrelated components for more efficient processing. Existing signal composition techniques of WiFi sensing are mostly for combining only CSI from multiple devices. New schemes are needed to integrate CSI with signals and information from other domains. It is also important to balance signal compression and composition for efficient and effective WiFi sensing.

New WiFi sensing algorithms are also required to take full advantage of multi-domain information with time, spatial, and user dependence. New coordination algorithms are necessary for extracting useful information from different domains. Since CSI has some unique properties such as low spatial resolution and sensitive to environmental changes, it is crucial for WiFi sensing algorithms to be robust in different scenarios. Most existing deep learning solutions of WiFi sensing reuse DNNs for images and videos. It is necessary to find suitable DNN types and develop new DNNs specifically designed for CSI data. For cross-sensor WiFi sensing, pre-trained DNNs for other sensors can be used for automatic labeling of CSI data. Transfer learning, teacher-student network training, and reinforcement learning can also be used to reduce network training efforts. WiFi sensing is very easy to be used for malicious purposes, since WiFi signals can be passively transmitted through walls and are not limited to lighting conditions. Generative Adversarial Networks (GANs) [41, 40] can be used to generate fake WiFi signal patterns to prevent from malicious WiFi sensing.

2.8 Chapter Summary

This chapter gives a survey of signal processing techniques, algorithms, applications, and performance results of WiFi sensing with CSI. It presents the basic concepts, advantages, limitations and use cases of the signal processing techniques and algorithms for different WiFi sensing applications. The survey highlights three WiFi sensing challenges: robustness and generalization, privacy and security, and coexistence of WiFi sensing and networking. Finally, the survey presents three future trends: integrating cross-layer network stack, multi-device cooperation, and fusion of different sensors, for improving existing WiFi sensing applications and enabling new sensing opportunities.

Chapter 3

SignFi: Sign Language

Recognition Using WiFi

3.1 Introduction

According to the World Federation of the Deaf (WFD), there are 70 million deaf people using sign language as their first language; many hearing people also use sign language as their first or second language¹. In the U.S. alone, there are one half to two million people using American Sign Language (ASL) in the 1990s [77]. Many colleges accept ASL as a foreign language credit, and more people are learning and using ASL. Modern Language Association conducted a survey of course enrollments in languages other than English from 2,696 institutions in the U.S. [37]. According to the survey, the number of ASL enrollments is consistently increasing from year 2002 to 2013. There are 109,577 ASL enrollments at 2013, making ASL the language with the third most enrollments.

There is a huge barrier between the Deaf community and people that do not understand or know little about sign language. A sign language recognition system would help break this barrier. There are some sign language recognition systems using cameras [138] or Kinect [210, 142, 116, 56], but they are subject to lighting conditions. Some systems use

¹<https://wfdeaf.org>

Leap Motion [26, 104, 124, 21, 105, 31, 97, 142, 210], but they can recognize only finger gestures and are very sensitive to the distance and displacement of the Leap Motion sensor and the human. Some systems use gloves and motion sensors, like SignAloud [107], but they are intrusive and require sensors to be attached on fingers.

Many papers have shown that Channel State Information (CSI) can be used to recognize hand [118, 2, 100, 144, 154, 136] and finger [79, 146, 219, 100] gestures in a non-intrusive way. WiFi signals are used to recognize ASL gestures in [136, 79, 100]. These are the most relevant papers to our work. But they are only evaluated on simple ASL gestures: 5 hand gestures in [136], 9 finger postures in [79], and 25 hand/finger gestures in [100]. Our object is to recognize nearly 300 basic sign gestures [141] that are frequently used in daily life. *The recognition algorithm should have high accuracy and low cost during testing.*

Classification algorithms of existing sign language recognition technologies have very low recognition accuracy when the number of sign gestures increases to nearly 300. Both papers [79, 100] use k-Nearest Neighbor (kNN) with Dynamic Time Wrapping (DTW) as the classification algorithm. We test it in a lab environment using CSI traces of 276 sign gestures. The average recognition accuracy of kNN with DTW is only 68% for 276 sign gestures. Moreover, kNN with DTW takes extremely long time in the testing stage when there are 276 classes. Thus, new classification algorithms are needed for sign gesture recognition using WiFi.

We propose SignFi to accurately recognize sign gestures using a 9-layer Convolutional Neural Network (CNN). It collects CSI measurements to capture wireless signal characteristics of sign gestures. After removing noises, SignFi feeds the pre-processed CSI measurements to a 9-layer CNN for sign gesture classification. We collect CSI traces for 276 sign gestures, each with 20 instances for the lab environment and 10 instances for the home environment. The average recognition accuracy of SignFi is 98.01%, 98.91%, and 94.81% for the lab, home, and lab+home environment, respectively. Fig. 3.1 compares SignFi with existing sign language recognition technologies. Most of the existing technolo-

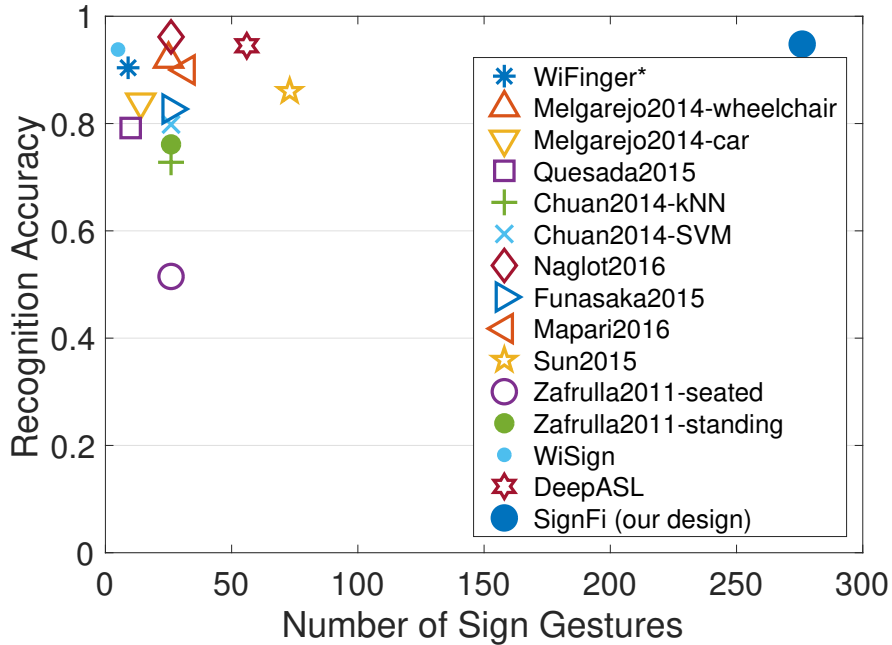


Figure 3.1: Comparison of different sign language recognition technologies. More details are shown in Table 3.1.

gies are tested on simple ASL gestures, such as 9 digital numbers and 26 alphabet letters. SignFi is the only one that is able to recognize 276 sign gestures with 94.81% accuracy. In summary, we make the following contributions:

- We propose a new signal processing technique to remove noises from raw CSI measurements. The information about how CSI changes over sub-carriers and sampling time is recovered.
- We present a 9-layer Convolutional Neural Network for accurate sign gesture recognition using WiFi signals.
- Our design has above 94% accuracy for 8,280 instances of 276 sign gestures from lab and home environments. We also run tests on 7,500 instances of 150 sign gestures from 5 different users and get 86.66% accuracy.

The rest of the chapter is organized as follows. §3.2 summaries existing sign recognition technologies and compares them with SignFi. §3.3 gives the motivation of sign language recognition using WiFi signals. The SignFi design, including signal processing and a 9-layer CNN, is presented in §3.4. §3.5 shows experiment setup and evaluation results. §3.6 summaries the chapter.

3.2 Related Work

There are many sign language recognition systems using different signals. We give a comparison of different sign language recognition technologies in Table 3.1. Since sign language recognition needs gesture recognition, we give a summary of gesture recognition technologies in Table 3.2.

3.2.1 Sign Language Recognition

A brief summary of sign language recognition technologies is given in Table 3.1. There are many vision-based sign language recognition systems using cameras [138] or the Kinect sensor [210, 142, 116, 56]. For example, the SignAll prototype [138] uses three cameras and one depth sensor to track hand gestures. A Kinect sensor, along with color gloves and accelerometers, are used to recognize 26 alphabet letters in [210]. Only the Kinect sensor is used in [142, 116, 56] to recognize sign gestures. Paper [142] is able to recognize 73 basic signs with 86.0% accuracy. These vision-based systems are subject to lighting conditions. Recently, many papers use the Leap Motion sensor for sign language recognition [104, 124, 21, 105, 31, 97, 142, 210]. As shown in Table 3.1, these systems are tested on signs for numbers and alphabet letters. These signs only involve simple finger postures. Leap Motion-based systems can only recognize finger gestures, and the hands must be in a very small region near the sensor. Some sign recognition systems use sensors, like motion sensors in SignAloud [107] and surface Electromyography (sEMG) sensors in [132], but they are intrusive and require sensors to be attached on fingers.

Table 3.1: Comparison of sign language recognition technologies.

Comparison Technologies	Device Used	Intrusive?	Granularity	Gesture Type	Recognition Algorithm ^a	Number of Sign Gestures	Recognition Accuracy
Zafrulla2011 [210]	Kinect, gloves and sensors	Yes	Hand/Finger	Static	HMM	26	51.5% (seated); 76.12% (standing)
Sun2015 [142]	Kinect	No	Hand/Finger	Dynamic	SVM	73	86.0%
Pigou2015 [116]					CNN	20	91.7%
Huang2015 [56]					CNN	25	94.2%
Chuan2014 [21]	Leap Motion	No	Finger	Static	kNN; SVM	26	72.78% (kNN); 79.83% (SVM)
Quesada2015 [124]					SVM	10	79.17%
Funasaka2015 [31]					SVM	26	82.71%
Mapari2016 [97]					MLP	26	90%
Naglot2016 [105]					MLP+BP	26	96.15%
DeepASL [26]	Leap Motion	No	Hand/Finger	Dynamic	RNN	56	94.5%
Savur2015 [132]	sEMG Sensor	Yes	Finger	Static	SVM	26	91% (offline); 82.3% (real-time)
WiSign [136]	WiFi (2.4/5 GHz)	No	Hand	Dynamic	SVM	5	93.8%
WiFinger* [79]			Finger	Static	kNN+DTW	9	90.4%
Melgarejo2014 [100]			Hand/Finger	Dynamic	kNN+DTW	25 (wheelchair); 14 (car)	92% (wheelchair); 84% (car)
SignFi (Our design)	WiFi (5 GHz)	No	Head/Arm/ Hand/Finger	Dynamic	CNN	276	98% (lab); 98% (home); 94% (lab+home)

^a HMM: Hidden Markov Model; SVM: Support Vector Machine; CNN: Convolutional Neural Network; kNN: k Nearest Neighbor; MLP: Multi-Layer Perceptron; BP: Back-Propagation; RNN: Recurrent Neural Network; DTW: Dynamic Time Warping

WiFi signals are used to recognize sign gestures in a non-intrusive way in [136, 79, 100]. But they can only recognize simple ASL gestures: 5 hand gestures in [136], 9 digits finger postures in [79], and 25 hand/finger gestures in [100]. SignFi also uses WiFi signals, and it is able to recognize 276 very complex sign gestures with 97.03% accuracy. For all the existing systems in Table 3.1, only [142] is evaluated on a relatively large number of complex sign gestures. It is able to recognize 73 sign gestures with 86.0% accuracy. These 73 sign gestures do not include signs that look similar in vision. SignFi is able to distinguish more complex sign gestures that have very similar hand/arm/finger movements, and with higher accuracy than the Kinect-based solution in [142].

3.2.2 Gesture Recognition

One important part of sign language recognition is gesture recognition. Table 3.2 gives a comparison of gesture recognition technologies using different signals. Motion sensors, like accelerometers used in [111, 175], are widely used for hand gesture recognition. Some papers use accelerometers and gyroscopes to recognize finger gestures [179, 201]. A smart-watch with accelerometers and gyroscopes is able to measure tendons movements and identify 37 (13 finger, 14 hand and 10 arm) gestures with 98% accuracy [201]. However, finger gestures must have the wrist and arm affixed to the chair arm while hand gestures must have the arm affixed. So finger gestures involve only finger movements, and hand gestures involve only wrist movements. This is not realistic for sign gesture recognition wherein a sign gesture may contain all hand, arm and finger movements. Magnetic sensors are used in [13, 14, 15] and sEMG sensors are used in [132] to recognize finger gestures, but these methods require sensors attached to the fingers of the signer. Sensor-based gesture recognition systems are intrusive.

Many gesture recognition systems use audio or wireless signals as the input. Sound-Wave [45] and AudioGest [130] use audio signals to recognize simple hand gestures with up to 95% accuracy. Audio signals are used in [106, 209, 167] to track 2-D finger movements with tracking accuracy of 8mm, 1cm, and 4.6mm, respectively. For audio-based

Table 3.2: Comparison of gesture recognition technologies.

Comparison Technologies	Signal/Device Used	Intrusive?	Granularity	Number of Gestures ^a	Recognition Accuracy
E-Gesture [111]	Motion Sensor	Yes	Hand	8	94.6%
Watanabe2016 [175]			Hand	15	79%
Serendipity [179]			Finger	5	87%
Xu2015 [201]			Arm/Hand/Finger	37	98%
FingerPad [13]	Magnetic Sensor	Yes	Finger	N/A	N/A (finger tracking)
uTrack [14]					
Finexus [15]					
Savur2015 [132]	sEMG Sensor	Yes	Finger	26	91% (offline); 82.3% (real-time)
SoundWave [45]	Audio (18-22 KHz)	No	Hand	5	94.7% (home); 94.3% (cafe)
AudioGest [130]			Hand	6	94.15%
FingerIO [106]			Finger	N/A	N/A (finger tracking)
Strata [209]			Finger	N/A	N/A (finger tracking)
LLAP [167]			Hand/Finger	N/A	N/A (hand/finger tracking)
SlideSwipe [221]	GSM (850 MHz)	No	Hand	14	87.2%
AllSee [71]	TV (725 MHz); RFID (915 MHz)	No	Hand/Finger	8	94.4% (TV); 97% (RFID)
RF-IDraw[160]	RFID (922 MHz)	Yes	Finger	N/A	N/A (finger tracking)
WiSee [118]	WiFi (2.4/5 GHz)	No	Body/Hand/Leg	9	94%
WiDraw [144]			Hand	N/A	N/A (hand tracking)
WiGest [2]			Hand	7	87.5% (1 AP); 96% (3 APs)
WiAG [154]			Hand	6	91.4%
WiSign [136]			Hand	5	93.8%
WiFinger** [146]			Finger	8	93%
Mudra [219]			Finger	9	96%
WiFinger* [79]			Finger	9	90.4%
Melgarejo2014 [100]			Hand/Finger	25 (wheelchair); 14 (car)	92% (wheelchair); 84% (car)
Molchanov2015 [103]			FMCW (24 GHz)	No	Finger
Soli [83]	Millimeter Wave (60 GHz)	No	Finger	4	92.1%
SignFi (Our design)	WiFi (5 GHz)	No	Head/Arm/ Hand/Finger	276	98% (lab); 98% (home); 94% (lab+home)

^a Only WiSign [136], WiFinger* [79], Melgarejo2014 [100], and SignFi (our design) evaluate on sign language gestures.

gesture recognition, the distance between the device and the hand/finger must be very short (usually less than 20cm). There are many gesture recognition systems using wireless signals, including Global System for Mobile communications (GSM) [221], TV [71], Radio-Frequency Identification (RFID) [71, 160], WiFi [118, 2, 136, 154, 100, 79, 146, 219, 144], Frequency Modulated Continuous Wave (FMCW) [103], and millimeter wave [83]. Sign language recognition needs to distinguish finger-level gestures/postures, which are not tested by SlideSwipe [221], WiSee [118], WiDraw [144], WiGest [2], or WiAG [154]. Although other wireless-based gesture recognition systems can detect finger-level gestures, none of them are tested on more than 25 gestures. SignFi is able to recognize 276 sign gestures with above 94% accuracy using WiFi signals.

3.3 Motivation

Our object is to recognize nearly 300 basic ASL gestures that are frequently used in daily life using CSI. *The classification algorithm should have high recognition accuracy and low computational cost during testing.* K-Nearest Neighbor (kNN) with Dynamic Time Wrapping (DTW) is used in both [79] and [100] to recognize simple sign language gestures. The question is whether kNN with DTW still works when the number of sign gestures increases by one order of magnitude.

In terms of computational cost of testing, kNN with DTW is not efficient when there are nearly 300 possible classes. Although kNN with DTW takes no time to train, it has extremely high overhead at testing time. For each testing sample, kNN with DTW needs to compare it with every single training sample. This requires a lot of computation resources during testing when there are 276 possible classes and each training/testing sample has 3,600 data points. Even for only 25 sign gestures, kNN with DTW needs more than 5 seconds per sign gesture during testing, which is shown later in §3.5.2.

In terms of recognition accuracy, there are three challenges for recognizing nearly 300 sign gestures: (1) the number of sign gestures is large; (2) many different sign gestures

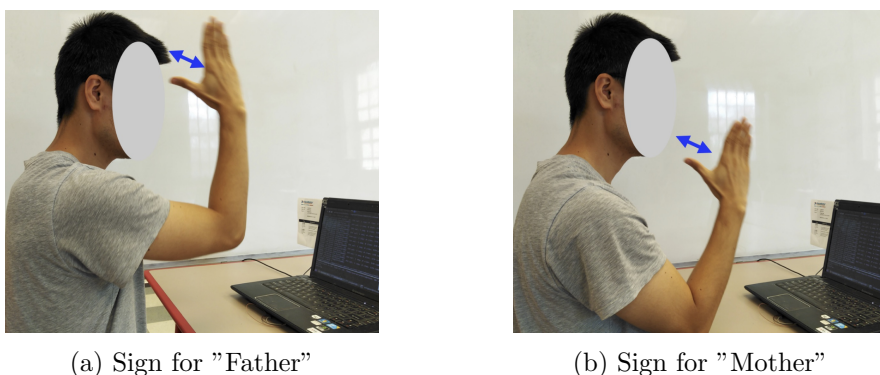
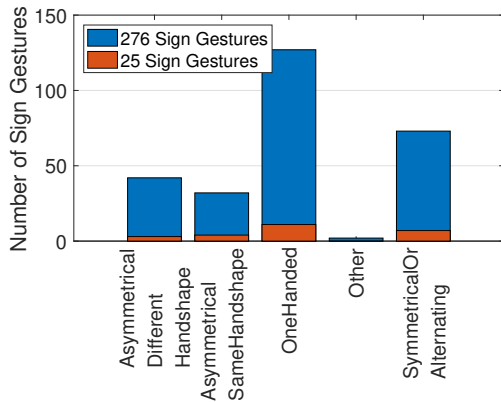


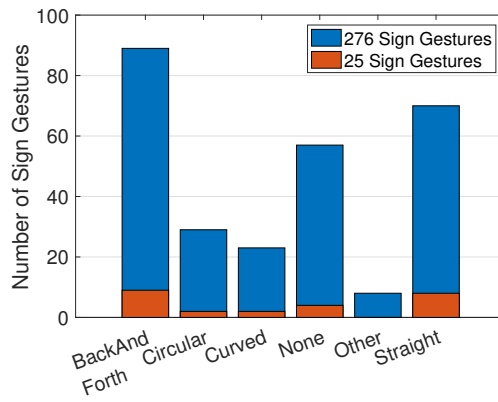
Figure 3.2: Sign words "Father" and "Mother" have the same hand gesture and finger posture, but they need the dominate hand in different locations.

have similar arm, hand, or finger movements; (3) many sign gestures involve complex and diverse movements. First, no more than 25 sign gestures are tested in [136, 79, 100]. There are nearly 300 basic sign words that are frequently used in daily life [141]. Second, since many sign words have similar gestures or postures, it is very hard to distinguish them from each other. For example, sign words "Father" and "Mother" have the same hand gesture and finger posture, but they require the dominant hand in different locations, as shown in Fig. 3.2. Finally, many sign gestures involve head, arm, hand, and finger movements. The dominant hand is not constrained in a small area; it can be near different parts of the human body.

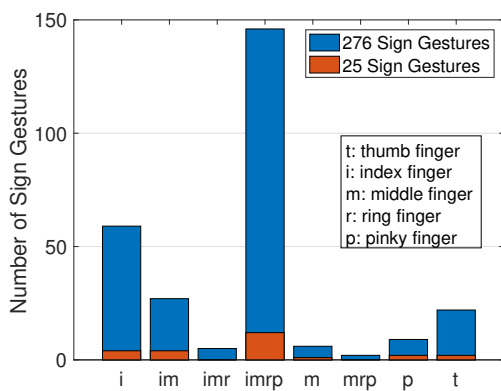
For the 276 sign gestures used in our experiments, we check their movement types using the ASL-LEX database [12]. The database gives sign types, path movement types, general locations, specific locations, and moving/foregrounded fingers of the dominant hand for nearly 1,000 sign gestures. We manually add the labels for gestures that are not included in ASL-LEX. Fig. 3.3 shows the number of sign gestures in each category. Many of the 25 sign gestures are evenly distributed in 3 or 4 categories, which makes them different from each other. For the 276 sign gestures, there are much more gestures in the same category, which makes them harder to be distinguished. For specific hand location, the 276 sign gestures have 12 categories that are not covered by the 25 sign gestures. Therefore, it is much harder to recognize the 276 sign gestures.



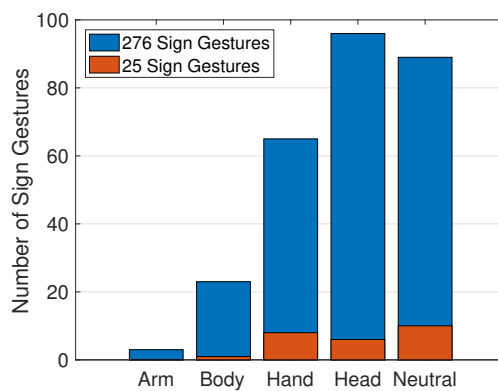
(a) Battison's Sign Types



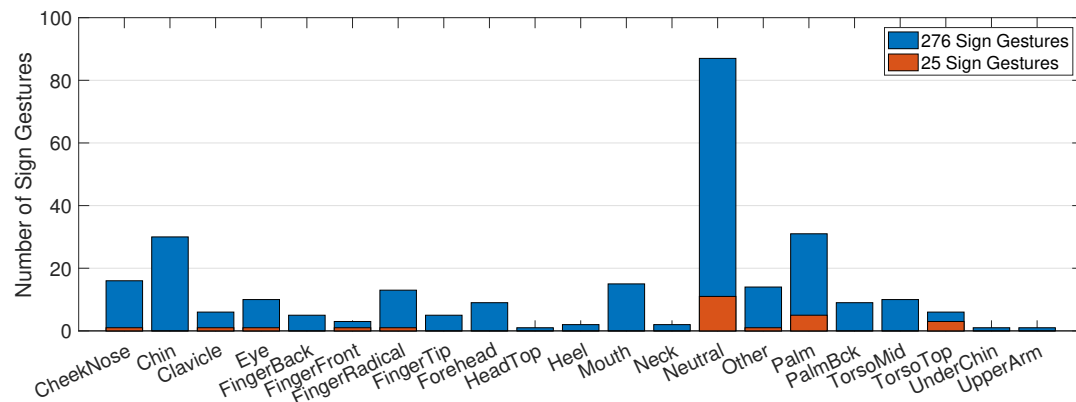
(b) Path Movement Types



(c) Moving/Foregrounded Fingers



(d) General Hand Locations



(e) Specific Hand Locations

Figure 3.3: Moving/foregrounded fingers, general and specific locations of the dominant hand of the 276 sign gestures used in our experiments. They involve more complex and diverse movements than the 25 sign gestures.

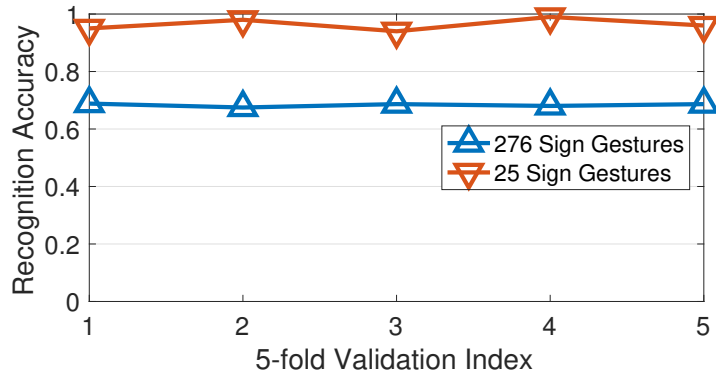


Figure 3.4: The average recognition accuracy of kNN with DTW decreases from 96% to 68% when the number of sign gestures increases from 25 to 276.

Can kNN with DTW still work for the 276 sign gestures? To answer this question, we collect CSI measurements and evaluate kNN with DTW in a lab environment. Fig. 3.4 shows the recognition accuracy of kNN with DTW for 25 and 276 sign gestures. For the 25 sign gestures, kNN with DTW has above 96% accuracy, which is in consistent with [79, 100]. However, the average accuracy drops to 68% for the 276 sign gestures. Thus, new algorithms are needed to improve recognition accuracy and reduce cost during testing for sign language accuracy using WiFi signals. For this purpose, we propose SignFi using a 9-layer CNN as the classification algorithm. It has high recognition accuracy and low cost during testing.

3.4 SignFi Design

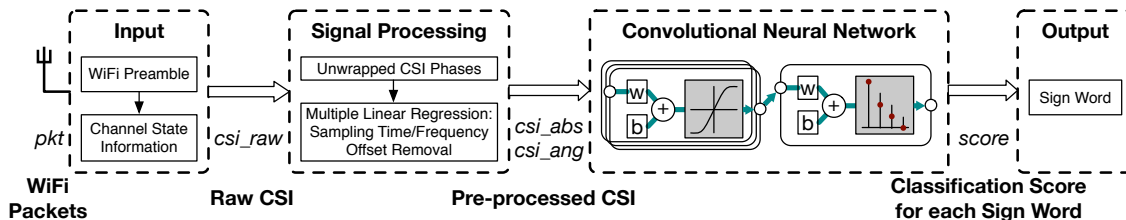


Figure 3.5: Overview of SignFi.

The SignFi overview is shown in Fig. 3.5. SignFi collects CSI measurements by WiFi

preambles. Raw CSI measurements are pre-processed to remove noises. Both amplitude and phase of the pre-processed CSI are fed to a 9-layer CNN for sign classification. In this paper, we mainly focus on the classification algorithm. CSI measurements are manually segmented for each sign gesture. We use the manually segmented CSI traces for both the proposed and existing classification algorithms for fair comparison. We leave automatic time segmentation to future work.

3.4.1 SignFi Signal Processing

We need to remove noises before feeding raw CSI measurements to the classification algorithm. In real-world WiFi systems, the sampling clocks and carrier frequencies of the transmitter and receiver are not synchronized. This leads to Sampling Time Offset (STO) and Sampling Frequency Offset (SFO) which introduce random phase shifts.

In our experiments, we use a linear antenna array with 3 transmit antennas. In this case, the CSI of the i th antenna is

$$h_i = h e^{-j2\pi(i-1)\Delta \cos \psi} = \left(\sum_n^N a_n e^{-j2\pi d_n/\lambda + j\phi_n} \right) e^{-j2\pi(i-1)\Delta \cos \psi}, \quad (3.1)$$

where h is the multi-path CSI in equation (3.1), Δ is the transmit antenna separation normalized to the unit of carrier wavelength, and ψ is the angle of departure with respect to the transmit antenna array [148]. CSI h_i captures the impact of multi-path channel propagation and the arrangement of transmit antenna array. The transmit antenna array adds the term $(i-1)\Delta \cos \psi$ to the CSI phase of the i th transmit antenna. We can rewrite the phase of h_i as $\angle h_i = \Phi_i - 2\pi(i-1)\Delta \cos \psi$, where Φ_i is the CSI phase caused by multi-path channel propagation. Our interest is to get the CSI phase $\angle h_i$ for each sub-carrier.

The measured CSI phase of the k th sub-carrier of the i th transmit antenna is

$$\Theta_{i,k} = \angle h_{i,k} - 2\pi f_\delta(k-1)\xi = \Phi_{i,k} - 2\pi(i-1)\Delta \cos \psi - 2\pi f_\delta(k-1)\xi, \quad (3.2)$$

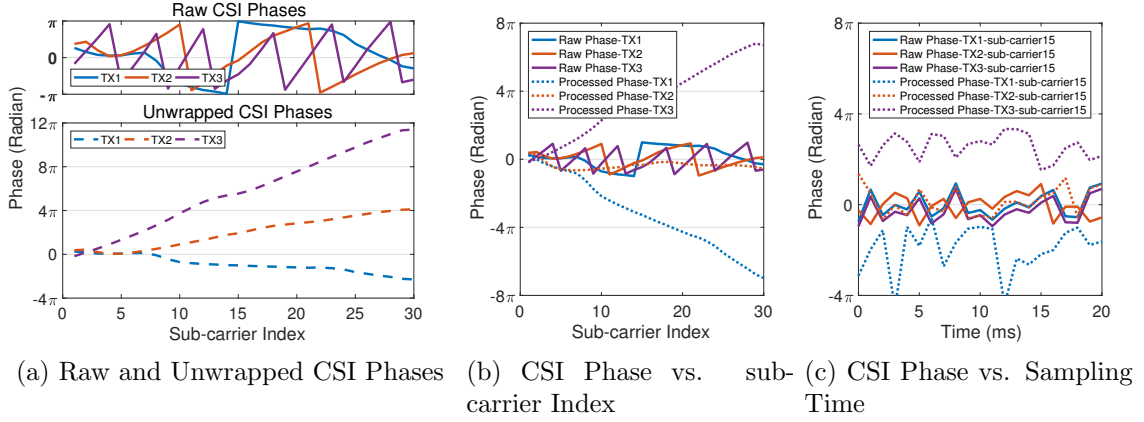


Figure 3.6: Raw CSI measurements do not capture how CSI phases change over sub-carriers and sampling time.

where $\Phi_{i,k}$ is the CSI phase caused by multi-path channel propagation, f_δ is the frequency spacing between two consecutive sub-carriers, and ξ is the phase offset caused by STO and SFO. As shown in Fig. 3.6a, the unwrapped CSI phases of each transmit antenna have different slopes caused by the term $(i-1)\Delta \cos \psi$ in equation (3.1). We estimate ξ by minimizing the linear fitting error across K sub-carriers and N transmit antennas

$$\hat{\xi} = \arg \min_{\omega} \sum_{i,k} (\Theta_{i,k} + 2\pi(i-1)\eta + 2\pi f_\delta(k-1)\omega + \beta)^2, \quad (3.3)$$

where η , ω and β are the fitting variables of multiple linear regression. The pre-processed CSI phase after removing random phase shifts is

$$\widehat{\mathcal{L}}h_{i,k} = \Theta_{i,k} + 2\pi f_\delta(k-1)\hat{\xi}. \quad (3.4)$$

Since the measured CSI phases are wrapped in the range of $[-\pi, \pi]$, raw CSI measurements give wrong information about how CSI phases change over sub-carriers and sampling time. As shown in Fig. 3.6b, raw CSI phases change periodically from $-\pi$ to π , while pre-processed CSI phases change nearly linearly in a wider range. Similarly, CSI phase variations over time are also corrected after CSI pre-processing. As shown in

Fig. 3.6c, raw CSI phases of the first and second transmitting antenna change similarly, but they have very different changing patterns for pre-processed CSI phases. Raw CSI phases give redundant information, and the pre-processed CSI phases recover the information about how CSI phases change over sub-carriers and sampling time. The pre-processed CSI phases can be used by other CSI-based sensing applications.

3.4.2 Gesture Recognition Algorithm

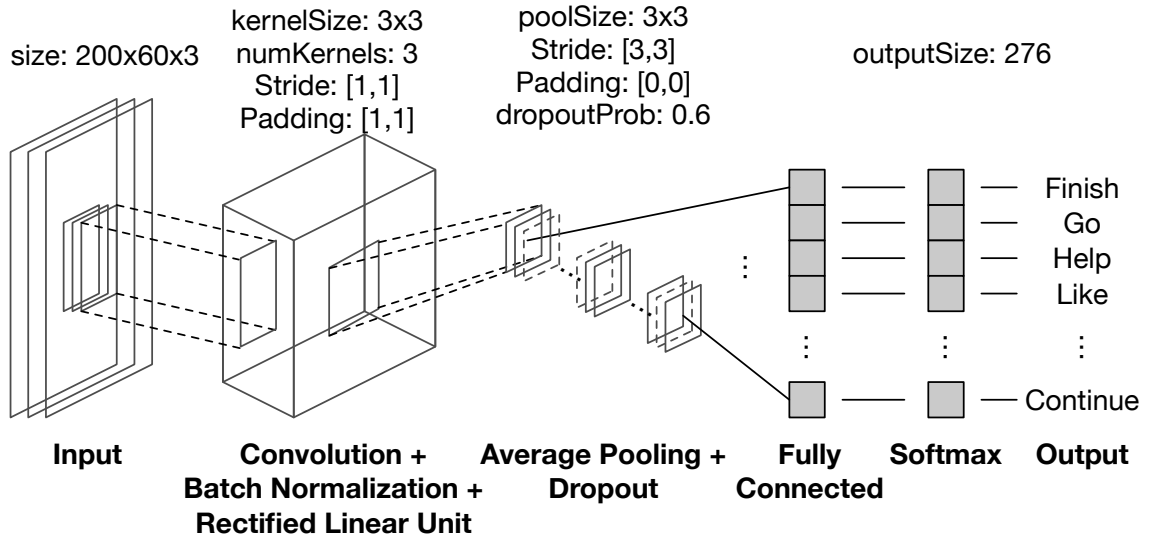


Figure 3.7: Neural architecture and parameter settings of the 9-layer CNN of SignFi.

SignFi uses a 9-layer CNN as the classification algorithm. CNNs are able to automatically learn parameters and features to find effective solutions for complex problems. Besides, CNNs are very fast to run in the inference stage even when the number of classes is very large. A neural network can be organized into multiple layers. The i th layer of a n -layer neural network is given by

$$\mathbf{y}^{(i)} = g^{(i)} \left(\mathbf{W}^{(i)} \mathbf{x}^{(i)} + \mathbf{b}^{(i)} \right), \quad (3.5)$$

where $\mathbf{y}^{(i)}$ is the output, $\mathbf{x}^{(i)}$ is the input, $\mathbf{W}^{(i)}$ is the weight matrix, $\mathbf{b}^{(i)}$ is the bias vector, and $g^{(i)}$ is the activation function [40]. The output of the previous layer is the input of

the current layer, i.e., $\mathbf{x}^{(i)} = \mathbf{y}^{(i-1)}$. For the first layer, $\mathbf{x}^{(1)} = \mathbf{x}$ is the original input. For the last layer, $\mathbf{y}^{(n)} = \mathbf{y}$ is the final output. For classification problems, \mathbf{y} contains labels in corresponding to the input \mathbf{x} . A CNN is simply a neural network with at least one of its layers involving convolution operations. Fig. 3.7 shows the architecture and parameter settings of the 9-layer CNN used in SignFi.

Neural networks learn the weights \mathbf{W} and biases \mathbf{b} , using an optimization algorithm, at each layer to minimize the cost function. SignFi uses Stochastic Gradient Descent with Momentum (SGDM) to update the weights and biases. It takes small steps in the direction of the negative gradient of the loss function:

$$\theta_{l+1} = \theta_l - \alpha \nabla E(\theta_l) + \gamma(\theta_{l+1} - \theta_l), \quad (3.6)$$

where θ is the parameter vector, l is the iteration index, α is the learning rate, $E(\theta)$ is the loss function, and γ is the momentum term [40]. The momentum term γ controls the contribution of the previous gradient step to the current iteration. SignFi uses a momentum term of 0.9 and a learning rate of 0.01. To prevent overfitting, SignFi uses L2 regularization to add a regularization term for the weights to the loss function $E(\theta)$. The regularized loss function is

$$E_R(\theta) = E(\theta) + \lambda \Omega(\mathbf{W}), \quad (3.7)$$

where λ is the regularization factor, and $\Omega(\mathbf{W}) = \mathbf{W}^T \mathbf{W} / 2$ is the regularization function. The regularization factor of SignFi is 0.01.

Input Layer. The input layer converts pre-processed CSIs of each sign gesture into a multi-dimensional tensor, which is the input format required by the CNN. This layer does not learn any parameters; it just prepares data input for the following layers. For SignFi, the size of each CSI matrix is $size(csi) = (1, 3, 30)$. There are 200 CSI samples for each sign gesture, so the size of CSI trace for each sign gesture is $(3, 30, 200)$. The CSI amplitude and phase, each with size of $(3, 30, 200)$, of each sign gesture are combined and reshaped to a tensor of size $(200, 60, 3)$ by the input layer.

Convolutional Layer. The convolutional layer replaces matrix multiplications with convolution operations. SignFi uses two-dimensional convolution:

$$S(i, j) = (I * K)(i, j) = \sum_m \sum_n I(m, n)K(i - m, j - n), \quad (3.8)$$

where I is the input, and K is the kernel [40]. Fig. 3.8 shows an example of two-dimensional convolution with a 2×2 kernel. The convolutional layer divides the input into multiple regions. Within each region, it computes a dot product of the input with some weights. The matrix containing the weights is called a kernel. The convolutional layer goes through the input vertically and horizontally with the same kernel. The step size of the convolutional layer moves each time is called a stride. SignFi uses three 3×3 kernels and stride of 1 in both vertical and horizontal directions. To preserve the output size of the convolutional layer and ensure all inputs are used for the same number of times, SignFi uses a padding of 1 in both vertical and horizontal directions. It pads a column/row of zeros around the edges of the original input.

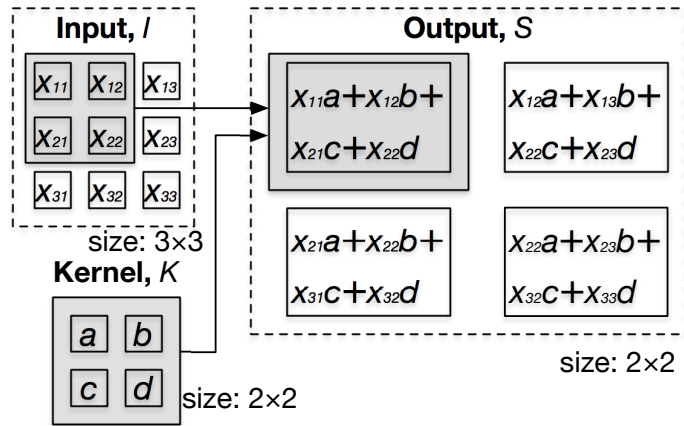


Figure 3.8: An example of two-dimensional convolution with a 2×2 kernel and stride of 1, reproduced from [40].

The number of kernels controls the number of channels in the output of the convolutional layer. For each input region, the convolutional layer adds a bias term to the dot product of the input and the kernel. The kernel, along with its bias term, is also called

a feature map. The convolutional layer learns the feature maps while going through the input. Since the convolution layer shares the same feature map for multiple input regions, it significantly reduces computation overhead for both training and testing. Convolutional layers are very effective and widely used in complex problems such as computer vision and natural language processing tasks. The impact of convolution on recognition accuracy of SignFi is shown later in §3.5.3.

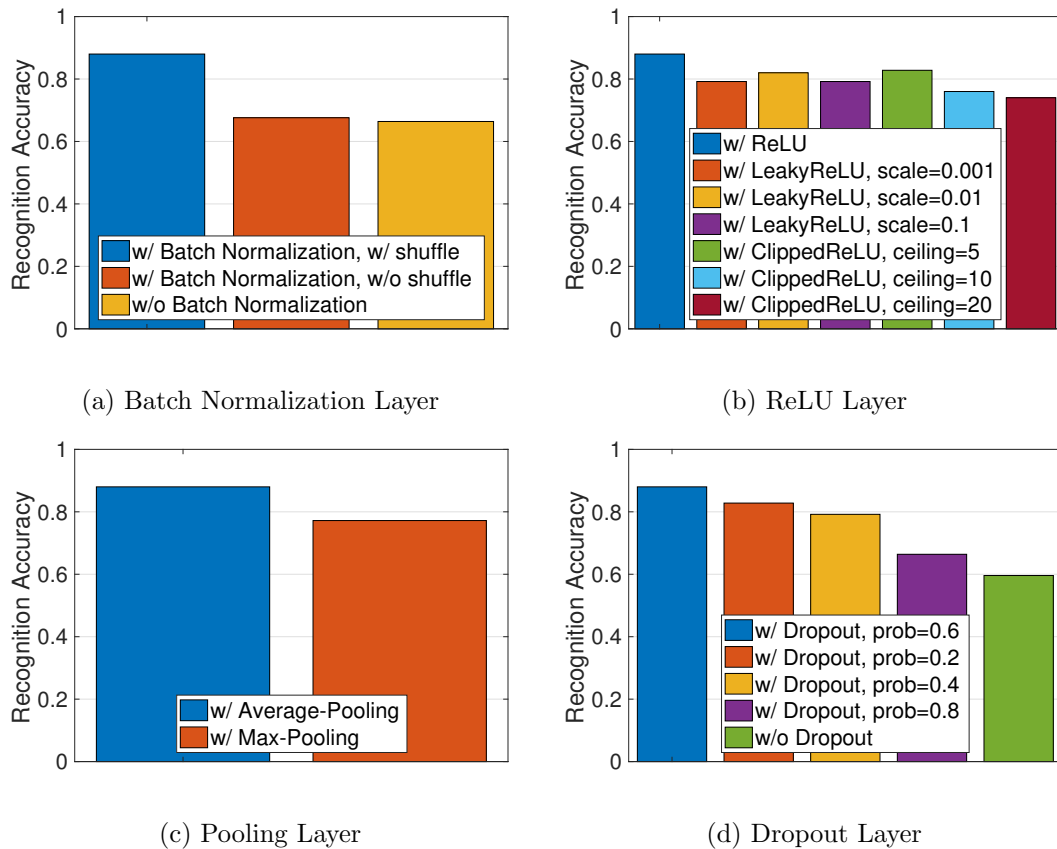


Figure 3.9: Impact of batch normalization, ReLU, pooling, and dropout on recognition accuracy using leave-one-subject-out validation for 25 sign gestures and 5 users.

Batch Normalization Layer. Batch normalization is used to speed up network training and reduce the sensitivity to network initialization. It makes the optimization problem easier. This allows a larger learning rate, making the network training much faster. It also improves generalization of the neural network when the training dataset

contains data from different users. It first normalizes its inputs x_i over a mini-batch. The normalized activation is

$$\hat{x}_i = \frac{x_i - \mu_B}{\sqrt{\sigma_B^2 + \epsilon}} \quad (3.9)$$

where μ_B and σ_B are the mean and variance of the mini-batch [63]. In case of near-zero variances, a very small number ϵ , which is 10^{-6} in SignFi, is used to improve numerical stability. The output of the batch normalization layer is $y_i = \kappa \hat{x}_i + \rho$, where κ is the scale factor, ρ is the offset, and \hat{x}_i is the normalized activation in equation (3.9) [63]. Both κ and ρ are learnable parameters that are updated during training. SignFi shuffles the training data after each training epoch to take full advantage of batch normalization

Fig. 3.9a shows the impact of batch normalization on recognition accuracy. The batch normalization layer helps prevent overfitting when the network sees a new user’s data that is not shown in the training stage. Without batch normalization, the neural network tends to overfit: the recognition accuracy is only 66% while training accuracy is nearly 100%. With batch normalization but without shuffling the training data, the recognition accuracy only improves by 1%. Batch normalization along with shuffling improves recognition accuracy by 22%.

Rectified Linear Unit Layer. The Rectified Linear Unit (ReLU) layer provides fast and effective training for deep neural networks, since its activation function is easy to compute and optimize. It has been shown more effective than traditional activations, such as logistic sigmoid and hyperbolic tangent, and is widely used in CNNs [40]. The ReLU layer performs a threshold operation to each input, where any input value less than zero is set to zero, as shown in equation (3.10). The size of the input is not changed after the ReLU layer.

ReLU: $g(x) = \begin{cases} x & \text{if } x \geq 0 \\ 0 & \text{if } x < 0 \end{cases} \quad (3.10)$	Leaky ReLU: $g(x) = \begin{cases} x & \text{if } x \geq 0 \\ scale * x & \text{if } x < 0 \end{cases} \quad (3.11)$	Clipped ReLU: $g(x) = \begin{cases} \tau & \text{if } x > \tau \\ x & \text{if } 0 \leq x \leq \tau \\ 0 & \text{if } x < 0 \end{cases} \quad (3.12)$
--	--	--

There are some modified ReLUs, like leaky ReLU in equation (3.11) and clipped ReLU in equation (3.12), but they have lower recognition accuracy than ReLU in our experiments. As shown in Fig. 3.9b, ReLU has 6% to 9% higher accuracy than leaky ReLU and 6% to 14% higher accuracy than clipped ReLU. The possible reason is that leaky ReLU introduces some noises when $x < 0$ and clipped ReLU loses some useful information when $x > \tau$ where τ is the clipped threshold.

Average-Pooling Layer. The average-pooling layer reduces the number of connections to the following layers by down-sampling. It returns the average of the inputs within a rectangular region. The pooling size of SignFi is 3×3 . Since there is no weight or bias, it does not provide any learning abilities. The major goal of average-pooling is to reduce the number of parameters to be learned in the following layers. It also helps reduce overfitting. Max-pooling returns the maximum, instead of the average, of selected inputs, but it has 10% lower recognition accuracy than average-pooling, as shown in Fig. 3.9c.

The convolutional layer, ReLU layer, and average-pooling layer are usually combined into one unit. There could be multiple of these units connecting with each other for large and complex datasets. We tried two and three of these units in our experiments, but get much lower recognition accuracy than using one unit.

Dropout Layer. The dropout layer is used to prevent overfitting. It randomly replaces a portion of its inputs with zero. In other words, it drops some randomly selected inputs, with a given dropout probability, and all the corresponding connections during training. As shown in Fig. 3.9d, the dropout layer with dropout probability of 0.6 improves recognition accuracy from 59% to 88%. Fig. 3.10 shows an example of the training and

testing process for SignFi with and without dropout. SignFi without dropout tends to overfit, since the training accuracy reaches 100% while the testing accuracy remains around 50% and does not increase much after the 100th iteration. Similar to the average-pooling layer, the dropout layer does not provide any learning abilities.

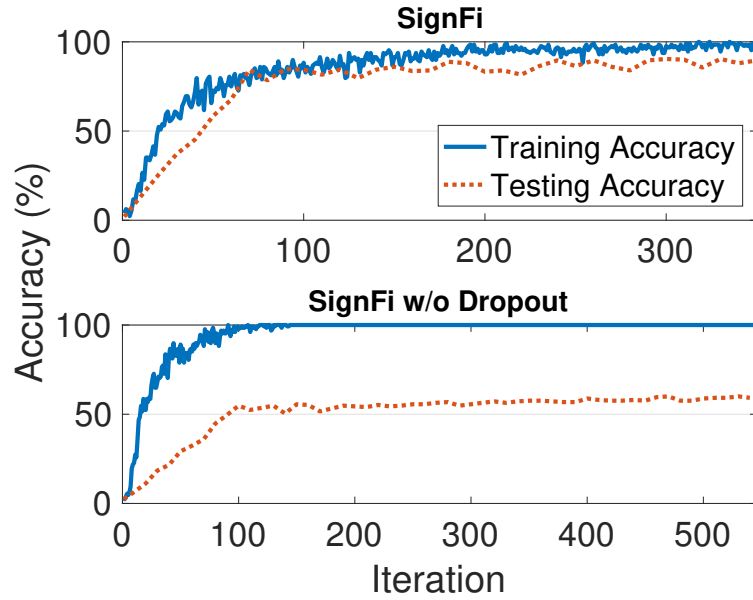


Figure 3.10: Training and testing accuracy. SignFi w/o dropout tends to overfit; there is a huge gap between training accuracy and testing accuracy.

Fully-Connected Layer. The fully-connected layer connects all of its neurons to the neurons in the previous layer, i.e., the dropout layer. The effect is to combine all the features learned by previous layers to classify the input. The size of fully-connected layer is equal to the number of all possible classes, i.e., 276 in our experiments.

Softmax Layer. A softmax layer and then a classification layer must follow the fully-connected layer for classification problems. The softmax layer applies the softmax function to the last fully connected layer:

$$P(c_r|x, \theta) = g(a(x, \theta))_r = \frac{e^{a_r(x, \theta)}}{\sum_{j=1}^k e^{a_j(x, \theta)}} = \frac{P(x, \theta|c_r)P(c_r)}{\sum_{j=1}^k P(x, \theta|c_j)P(c_j)}, \quad (3.13)$$

where $g(a)_r = e^{a_r} / \sum_{j=1}^k e^{a_j}$ is the softmax function with $0 \leq g(a)_r \leq 1$ and $\sum_{j=1}^k g(a)_j = 1$. Moreover, $a_r(x, \theta) = \ln(P(x, \theta|c_r)P(c_r))$, where $P(x, \theta|c_r)$ is the conditional probability of the given class r , $P(c_r)$ is the class prior probability, and θ is the parameter vector.

Classification Layer. The classification output layer takes the values from the softmax function and assigns each input to one of the k mutually exclusive classes using the cross entropy function

$$E(\theta) = - \sum_{i=1}^n \sum_{j=1}^k t_{ij} \ln y_j(x_i, \theta), \quad (3.14)$$

where t_{ij} represents that the i th sample belongs to the j th class, and θ is the parameter vector. $y_j(x_i, \theta)$ is the output for the i th sample, which is the value from the softmax function. It represents the probability that the network associates the i th input with class j , i.e., $P(t_j = 1|x_i)$.

3.5 Evaluation

In this section, we first give the experiment setup, including measurement layout and displacements, data collection procedure, and WiFi settings. We compare SignFi with existing classification algorithms in different environments. Two performance metrics, recognition accuracy and time consumption of training and testing, are evaluated. We also check the impact of convolution, signal processing, and sampling rate on the recognition accuracy of SignFi. Finally, we run user independence test for 150 sign gestures performed by 5 different users.

3.5.1 Experiment Setup

We collect CSI traces for 276 sign gestures that are frequently used in daily life. CSI traces are measured in both lab and home environments. Fig. 3.11 shows the measurement settings for the lab and home environments. The dimension of the lab and home is 13m×12m and 4.11m×3.86m, respectively. The lab has more surrounding objects, leading to a more complex multi-path environment than the home. The distance between the AP

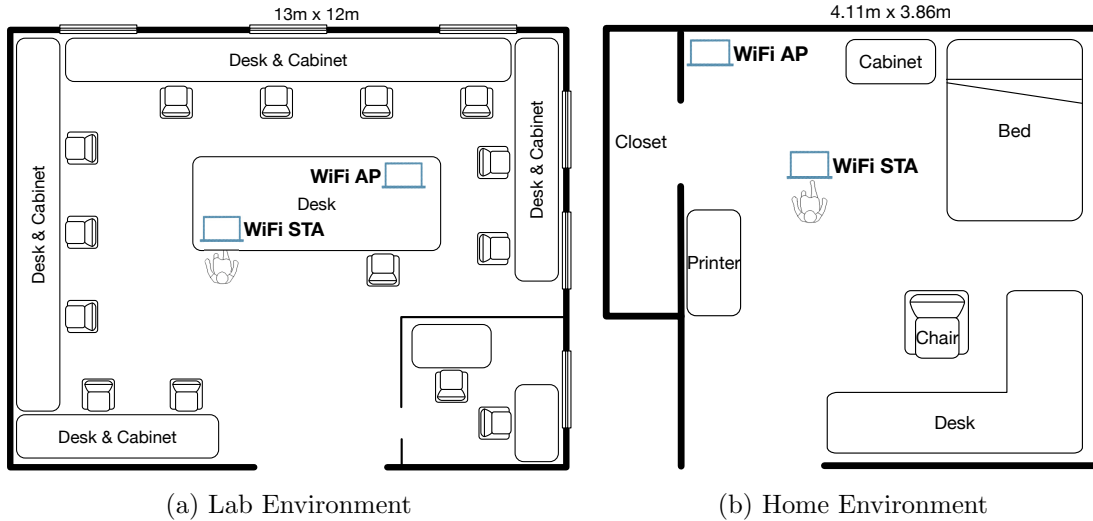


Figure 3.11: Floor plan and measurement settings of the lab and home environments.

and STA is 230cm and 130cm, respectively, for the lab and home environment. For the home environment, the transmit antenna array is orthogonal to the direction from the AP to STA. For the lab environment, the angle between the transmit antenna array and the direct path is about 40 degrees. The major differences of these two environments are: (1) dimension of the room, (2) distance between the AP and STA, (3) angle between the transmit antenna array and the direct path, and (4) multi-path environments.

Fig. 3.12 shows the experiment setup in the lab environment. During the experiments, each user first watches video on [141] to learn how to sign for one word. As long as the user feels comfortable to conduct the sign gesture smoothly, we begin to collect CSI traces for this sign gesture. The user repeats the sign gesture in front of a WiFi STA, which exchanges packets with a nearby WiFi AP. The user begins to make the first sign as seeing “Sign Starts ... 1” on the screen of the STA. At the same time, the AP sends 802.11n packets periodically to the STA. The STA collects CSI measurements while the person is making sign gestures. The user repeats the same sign gesture until the screen of the STA shows “Sign Starts ... [n]”. Here n could be 11 or 21 depending on whether 10 or 20 gesture instances are collected. We repeat this procedure, i.e., watching the video, repeating the sign gesture, and collecting CSI traces, for all the sign gestures.

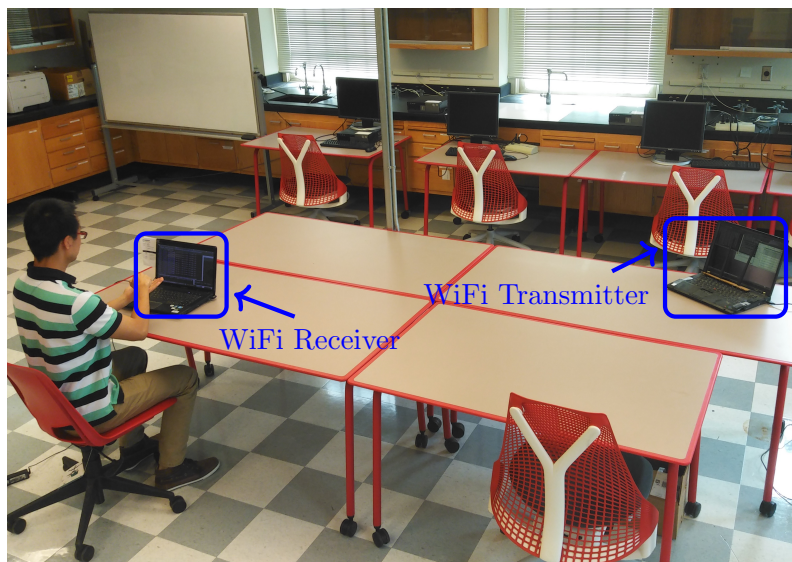


Figure 3.12: Experiment setup for the lab environment.

Table 3.3: Number of sign words in different categories used in the experiments.

Common	Animals	Colors	Descriptions	Family	Food	Home	People	Questions	School	Time	Others	Total
16	15	12	32	31	54	17	13	6	26	31	23	276

Table 3.3 summarizes the 276 sign words used in our experiments divided into different categories. We select the 253 basic sign words from [141]. These sign words are the most important words for ASL beginners and are frequently used in daily life. Some sign words have different gestures; we only select one of the gestures that have the same meaning. We do not select compound signs that are composed of more than three signs. For comparison, we also run experiments on 25 sign gestures from [100]. Two sign words, “phone” and “kitchen”, are already included in the 253 basic sign gestures. In total, 276 sign gestures are tested in our experiments.

The WiFi AP and STA are two laptops with Intel WiFi Link 5300 installed. CSI measurements are collected using openrf [76], which is modified based on the 802.11n CSI tool [51]. The WiFi AP and STA operate at 5GHz, and the channel width is 20MHz. Note that the 802.11n CSI tool only provides CSI values of 30 sub-carriers even though a 20MHz WiFi channel has 52 sub-carriers. The AP has 3 external antennas, and the STA

has 1 internal antenna. The transmitting power is fixed at 15dBm. All experiments are conducted in the presence of other WiFi signals. As shown in Fig. 3.12, the user is not on the direct path between the STA and AP. This is a normal case in real life. It also makes sign gesture recognition hard, since the strength of reflected signals is much lower than that of the direct path signals. Training and testing are performed by a Linux desktop with an 8-core i7-4790 CPU at 3.60GHz and 15.6GB of RAM.

Table 3.4: Data collection summary.

	User	Age	Weight/Height	Data Collection Date (# Signs \times # Repetitions)	Gesture Duration	# Instances
Lab	User 1	39	90kg/170cm	Oct. 18, 2017 (25×10); Nov. 2, 2017 (125×10)	1s-2.5s	1,500
	User 2	28	61kg/174cm	Oct. 18, 2017 (25×10); Oct. 30, 2017 (125×10)	0.5s-1.5s	1,500
	User 3	31	55kg/168cm	Oct. 21, 2017 (25×10); Nov. 6, 2017 (125×10)	0.5s-1.5s	1,500
	User 4	26	65kg/180cm	Oct. 23, 2017 (25×10); Oct. 31, 2017 (125×10)	1s-2.5s	1,500
	User 5 ^a	29	68kg/171cm	Jul. 18, 2017 (166×20); Jul. 19, 2017 (110×20)	0.5s-1.5s	5,520
Home	User 5	29	68kg/171cm	Jun. 8, 2017 (32×10); Jun. 25, 2017 (68×10); Jul. 4, 2017 (100×10); Jul. 11, 2017 (25×10); Jul. 12, 2017 (51×10)	0.5s-1.5s	2,760

^a Compared with user 1 to 4, user 5 has different experiment settings, such as laptop displacement, surrounding objects, desk and chair arrangements, etc., even though they are in the same lab environment. The data collection time of user 5 is 3-4 months earlier than that of user 1 to 4, so it is hard to recover the same settings.

We collect CSI traces from 5 male users who do not know how to sign before the experiments. Table 3.4 shows the summary of data collection. Different users may have different gesture durations and slightly different hand/finger movements for the same sign word. For user 1 to 4, we collect CSI traces only in the lab environment. Each of the 4 users makes 150 sign gestures with each gesture repeated for 10 times. There are 6,000 gesture instances for these 4 users. For user 5, we collect CSI traces in both the lab and

home environments. Each sign gesture has 20 instances for the lab environment and 10 instances for the home environment. In total, there are 8,280 gesture instances for user 5. CSI traces, labels, and videos of the 276 sign words are available for download².

3.5.2 Comparing SignFi with Existing Methods

We compare SignFi with the classification algorithm kNN with DTW used in [79, 100]. The input signals used in [100] are the Received Signal Strength (RSS) and the amplitude and phase of 5 sub-carriers with the least average cross-correlation. We feed the same input signals to kNN with DTW. CSI traces of different transmit antennas provides different results for kNN with DTW. We select the transmit antenna that has the highest recognition accuracy. We also use CSI traces from all the transmit antennas and all the 30 sub-carriers as input signals. To check the impact of input signals on CNN, we also run CNN using CSI traces from 5 or 30 sub-carriers. Table 3.5 gives a summary of the classification algorithm and CSI size for each label used in our comparison. In this section, we only use the data of user 5. We run 5-fold cross validation using 5,520 instances from the lab, 2,760 instances from the home, and 8,280 instances from the lap+home environment.

Table 3.5: CSI size of different recognition algorithms (Fig. 3.13, 3.14, and 3.15).

Label	Algorithm	CSI Size
kNN+DTW+5subc	kNN+DTW	(1,1,5)
kNN+DTW+5subc+MIMO	kNN+DTW	(1,3,5)
kNN+DTW+30subc	kNN+DTW	(1,1,30)
kNN+DTW+30subc+MIMO	kNN+DTW	(1,3,30)
CNN+5subc	CNN	(1,1,5)
CNN+30subc	CNN	(1,1,30)
SignFi	CNN	(1,3,30)

Recognition Accuracy. Recognition accuracy is defined as the number of correctly classified instances divided by the number of all testing instances. Fig. 3.13 shows recogni-

²<https://yongsen.github.io/SignFi/>

tion accuracy results in different environments. SignFi provides high recognition accuracy for all the datasets. For the 276 sign gestures, the average recognition accuracy of SignFi is 98.01%, 98.91%, and 94.81% for the lab, home, and lap+home environment, respectively. With only 5 sub-carriers of non-MIMO CSI traces, CNN has average accuracy of 80.74%, 93.37%, and 83.00% for the lab, home, and lab+home environment, respectively. Using the same input, with either 5 or 30 sub-carriers of non-MIMO CSI traces, CNN has much higher accuracy than kNN with DTW in the lab and lab+home environments. For the lab+home environment, SignFi still has 94.81% accuracy even though the lab and home have very different experiment settings.

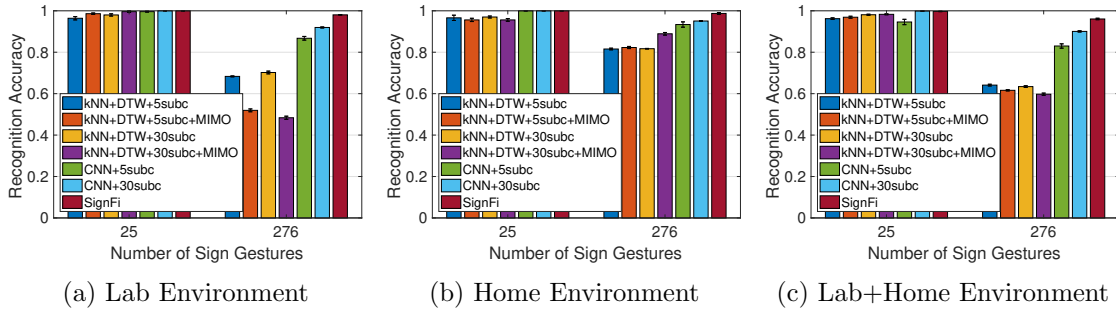


Figure 3.13: Recognition accuracy in different environments.

For the 25 sign gestures, all classification algorithms have over 95% recognition accuracy for all the three environment settings. For the 276 sign gestures, the accuracy of kNN with DTW decreases dramatically for the lab and lap+home environments. For the lab environment, the average recognition accuracy is only 68.33% and 70.20% for kNN with DTW using CSI traces of 5 and 30 sub-carriers, respectively. Adding MIMO CSIs further decreases the average accuracy to 51.93% and 48.30%. The major reason is that the lab has a complex multi-path environment, which heavily impacts MIMO. Another reason is that the distance between the WiFi AP and STA in the lab is longer than that of the home environment, so the strength of reflected signals is lower. This leads to more noise signals for the lab environment. The average accuracy of kNN with DTW is increased to 82% for the home environment. Using both MIMO and 30 sub-carriers of CSI traces, the accuracy

of kNN with DTW is improved to 88.90%. The reason is that the home environment has less multi-path signals and shorter distance between the AP and STA.

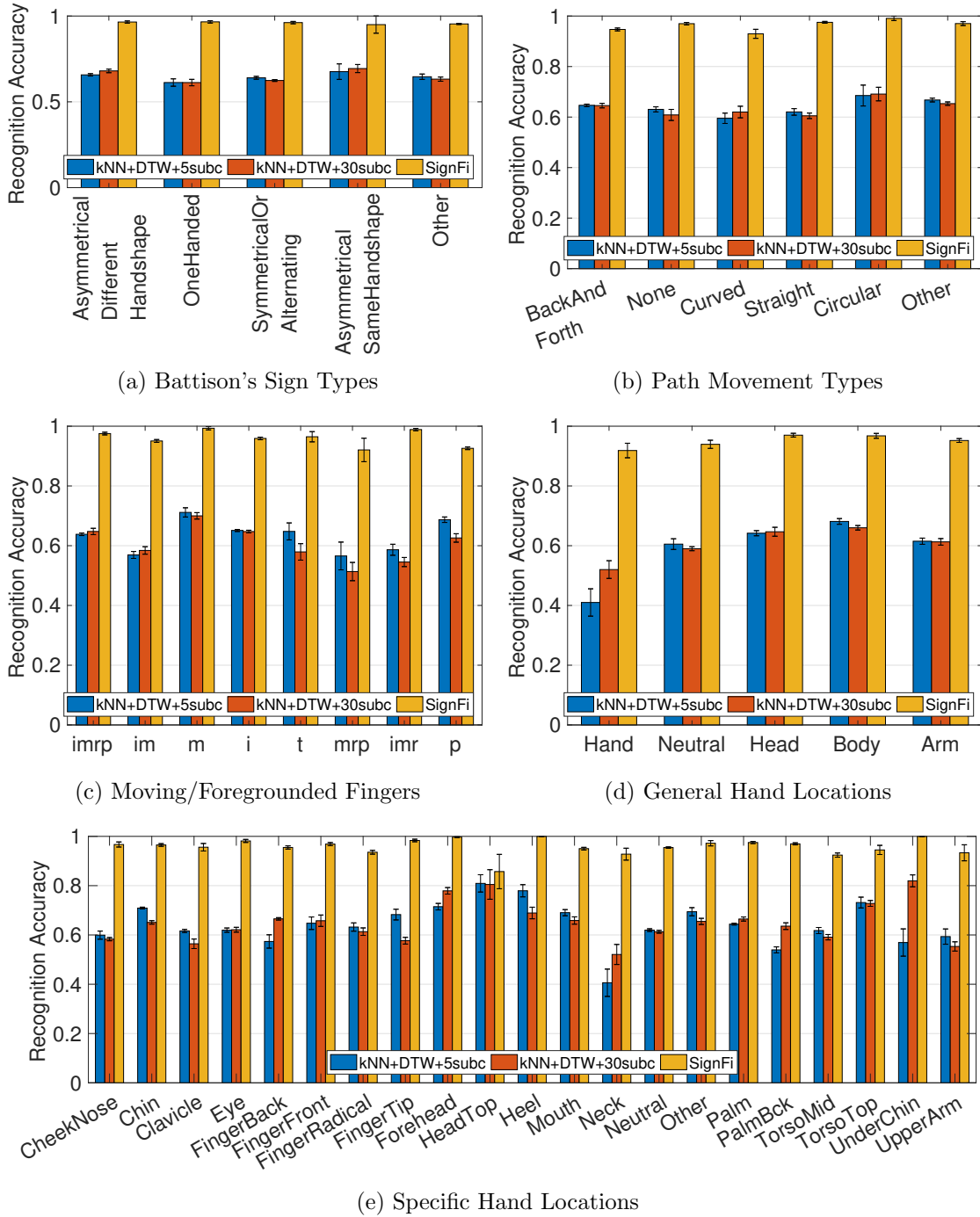


Figure 3.14: Recognition accuracy in different categories for the lab+home environment.

To get a better understanding about sign gesture recognition, we break down the 276 sign gestures into different categories and check the recognition accuracy in each category, as shown in Fig. 3.14. All the evaluation results in Fig. 3.14 are from the lab+home environment. The number of sign gestures in each group of each category for the 276 sign gestures is shown in Fig. 3.3 in §3.3. All the sign gestures in the same group of each category has similar patterns. The evaluation results show that: (1) whether the classification algorithms can distinguish similar sign gestures; (2) what kinds of sign gestures are hard to recognize. SignFi has high accuracy for all groups in each category. This means that even for sign gestures with very similar patterns, SignFi is still able to distinguish them from each other. For sign types in Fig. 3.14a and path movement types in Fig. 3.14b, there is no significant difference for each group. For Fig. 3.14c, kNN with DTW has the lowest accuracy when there are three moving/foregrounded fingers, “mrp” and “imr”. For general hand locations in Fig. 3.14d, sign gestures with the dominate hand near the non-dominate hand are the hardest to recognize, for both SignFi and kNN with DTW. For general hand location of “Hand” and specific hand location of “Neck”, the recognition accuracy of kNN with DTW is only 40% and 51% using 5 and 30 sub-carriers’ CSI, respectively. For specific hand location of “HeadTop”, kNN with DTW has comparable accuracy as SignFi, as shown in Fig. 3.14e. For all other groups in each category, kNN with DTW has much lower accuracy than SignFi.

Time Consumption of Training and Testing. Fig. 3.15 shows the time consumption of training and testing of different classification algorithms with different CSI inputs. The testing time of SignFi is much shorter than that of kNN with DTW. For kNN with DTW, the testing time of each sign gesture is 33.36ms using 5 sub-carriers of non-MIMO CSI traces. It proliferates to 5,147.20ms if the input has 30 sub-carriers of MIMO CSI traces. The testing time of SignFi is 0.62ms. However, SignFi does have longer training time than kNN with DTW. The maximum training time for kNN with DTW is only 0.046ms. SignFi takes 8.28ms for CSI processing and CNN training for each sign gesture. It takes 0.05ms and 2.12ms for CNN to finish training using 5 and 30 sub-carriers

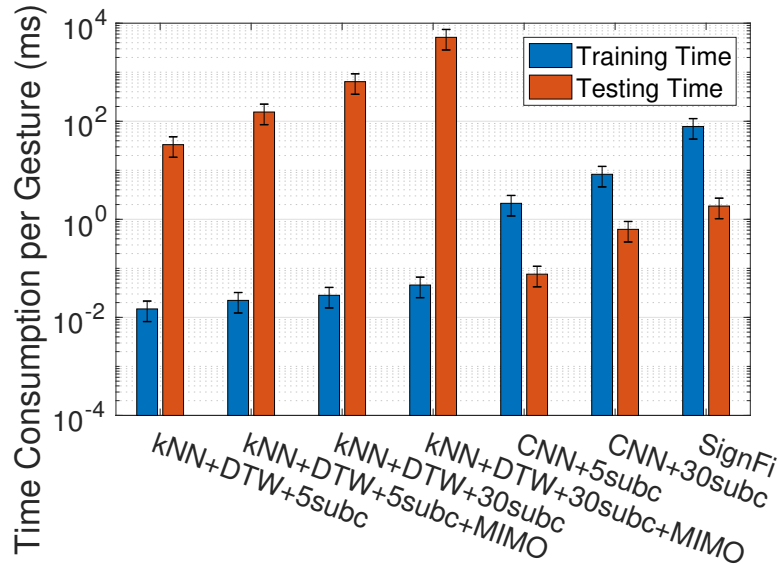


Figure 3.15: Time consumption of training and testing for each sign gesture.

of non-MIMO CSI traces, respectively. Since training usually can be performed offline and testing must be done in real-time, it is more important to reduce the testing time. Therefore, SignFi is more practical than kNN with DTW to be implemented in real-time.

3.5.3 More Discussions on SignFi

In this section, we investigate the impact of convolution, signal processing, and sampling rate on the recognition accuracy of SignFi. We run 5-fold cross validation using the data of 276 sign gestures from user 5.

Impact of Convolution. The reason we use CNN as the classification algorithm is that CNN has higher recognition accuracy than other neural networks that do not have the convolutional layer. As shown in Fig. 3.16a, the recognition accuracy of SignFi without convolution is 84.64%, 81.70%, and 70.34% for the lab, home, and lab+home environment, respectively. For the lab+home environment, the accuracy improvement due to convolution is 24.47%, which is the highest among all the three environments. The evaluation results show that the convolution layer has a significant impact on the

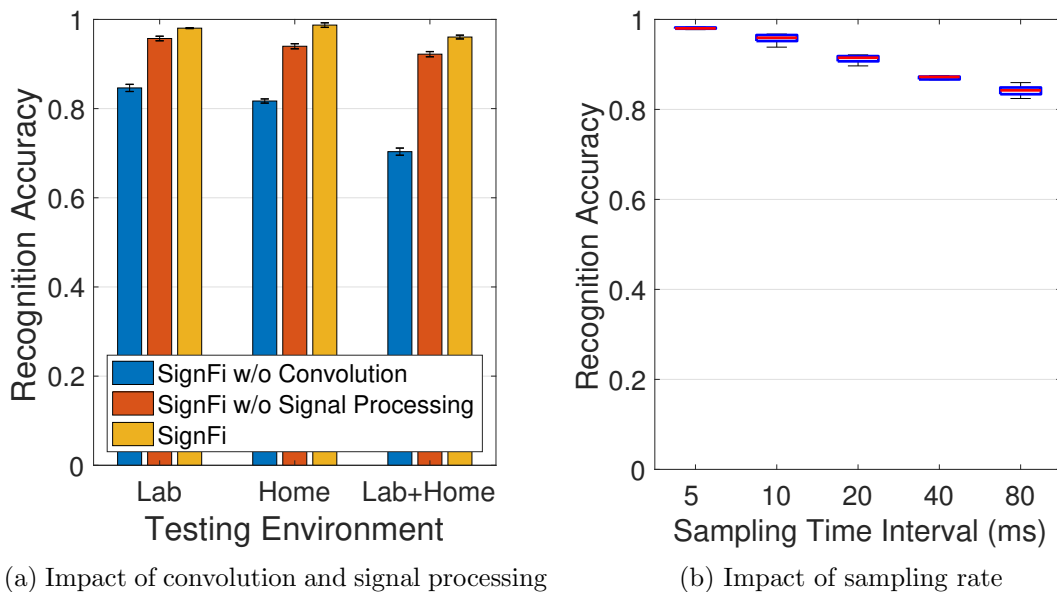


Figure 3.16: Impact of convolution, signal processing, and sampling rate of SignFi.

recognition accuracy of SignFi.

Impact of Signal Processing. We also check the impact of SignFi signal processing that removes noises from raw CSI measurements, as shown in Fig. 3.16. Without SignFi signal processing, the average recognition accuracy is 95.72%, 93.98%, and 92.21% for the lab, home, and lab+home environment, respectively. SignFi signal processing has the highest accuracy improvement, which is 4.93%, for the lab environment. The reason is that SignFi signal processing only removes random phase offsets. It does not filter out other noise signals. The layout, surrounding environment, and displacement of the AP and STA of the lab and home are very different, leading to very different noise signals.

Impact of Sampling Rate. Another import factor that influences the recognition accuracy of SignFi is the CSI sampling rate. For all the previous evaluation results, the WiFi STA measures CSI about every 5ms. We change the sampling rate and run SignFi with 5-fold cross validation for the CSI dataset of the lab environment. The evaluation results are shown in Fig. 3.16b. When the sampling time interval increases from 5ms to 10ms, the average recognition accuracy decreases from 98.01% to 95.72%. SignFi still has

high recognition accuracy using 10ms of sampling interval, considering there are 5,520 instances of 276 sign gestures. When the sampling time interval increases to 20ms, the average recognition accuracy decreases to 91.12%. Based on these results, the sampling time interval should be no larger than 20ms to get high recognition accuracy. For sampling interval of 40ms and 80ms, the average accuracy further decreases to 87.05% and 84.17%.

3.5.4 User Independence Test

This section gives user independence test using CSI traces of 150 sign gestures from 5 different users. There are 7,500 gesture instances in total. We run self test, 5-fold cross validation, and leave-one-subject-out validation. For self test, wherein training and testing only include CSI traces from the same user, the recognition accuracy for each user is above 98%, as shown in Fig. 3.17a. In §3.5.4, we use CSI traces of just user 1 to 4 to separate the impact of user 5. As shown in Table 3.4, CSI traces of user 5 were collected three to four months before user 1 to 4. We were unable to use the exact same experiment settings, such as laptop displacement, surrounding objects, desk and chair arrangements, etc., for user 1 to 4 and user 5. User 1 to 4 have almost the same experiment settings, which are different from that of user 5. We also show the impact of different data collection dates and settings by including CSI traces of user 5 in §3.5.4.

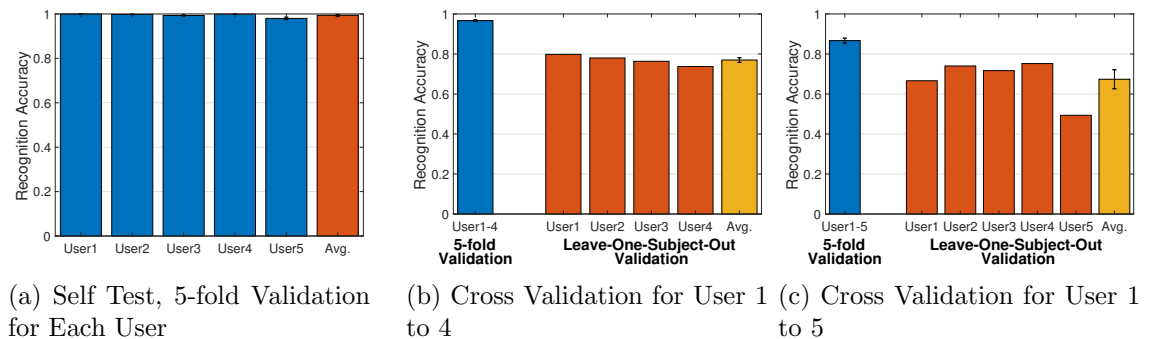


Figure 3.17: Recognition accuracy in different environments.

Users with Similar Data Collection Dates and Settings. We first run 5-fold cross validation using CSI traces of user 1 to 4. In other words, CSI traces from different

users are mixed together and randomly divided into training and testing datasets. The average recognition accuracy is 96.68%, as shown in Fig. 3.17b. This means that SignFi is robust to different users, considering these users have different body sizes and gesture durations. For leave-one-subject-out cross validation, the recognition accuracy is in the range of 73.73% and 79.80%, and the average recognition accuracy is 76.96%. Comparing with 5-fold cross validation, the recognition accuracy of leave-one-subject-out cross validation decreases by 20%.

Users with Different Data Collection Dates and Settings. When CSI traces of user 5 are included, the recognition accuracy of 5-fold cross validation decreases to 86.66%, as shown in Fig. 3.17c. This means that experiment settings have a significant impact on SignFi. For leave-one-subject-out validation, the average accuracy drops to 67.36%. The accuracy of user 1 to 3 decreases by 13.2%, 4%, and 4.7%, respectively. The recognition accuracy of user 5 is only 49.3%. This is not adequate for practical usage, but is still very good compared to a random algorithm, which only has $1/150=0.06\%$ accuracy. From the evaluation results, we can see that different users and different experiment settings have a great impact on SignFi. When there is a new user, we may need the user first conduct some sign gestures to train the neural network to get a good recognition accuracy.

3.6 Chapter Summary

This chapter presents a sign language recognition system, SignFi, to recognize frequently used sign gestures using WiFi signals. SignFi measures CSI by WiFi packets and uses a 9-layer CNN as the classification algorithm. The average recognition accuracy of SignFi is 98.01%, 98.91%, and 94.81% for the lab, home, and lab+home environment, respectively. For 7,500 instances of 150 sign gestures performed by 5 different users, the recognition accuracy of SignFi is 86.66%.

Chapter 4

Location and Person Independent Activity Recognition with WiFi, Deep Neural Networks and Reinforcement Learning

4.1 Introduction

In recent years, WiFi signals are widely used for non-intrusive sensing purposes. WiFi-based sensing applications are easy to deploy and have low costs by reusing the infrastructure that is designed for wireless communications. Multiple-Input Multiple-Output (MIMO) and Orthogonal Frequency-Division Multiplexing (OFDM) are two of the most important technologies to provide high performance for modern WiFi systems. MIMO-OFDM provides Channel State Information (CSI) which represents the power attenuation and phase shift from the transmitter to the receiver at certain carrier frequencies. In addition to improving the networking performance of WiFi networks, CSI can also be used for WiFi-based sensing applications since it captures how WiFi signals travel from the transmitter to the receiver through surrounding objects and hu-

mans. For example, when a person is moving or doing different activities around the WiFi transmitter or receiver, the reflected WiFi signals are changed accordingly. The CSI amplitude and phase are also impacted, and these CSI variations can be fed to pre-defined models or machine learning algorithms for human motion detection [157, 110, 52, 214, 216, 147, 42, 87, 38, 81, 30, 202, 89, 90, 5, 232, 91] and activity recognition [6, 23, 27, 30, 33, 43, 66, 158, 165, 166, 173, 170, 176, 189, 211, 25, 125, 192, 29].

For WiFi-based activity recognition to be practical in real-world scenarios, the recognition algorithm should be location and person independent. In the training stage, the recognition algorithm is usually trained in a controlled environment. During testing in real-world deployments, the location and orientation of WiFi devices are usually unknown and testing persons are unseen during training. However, it is challenging for WiFi-based activity recognition to be robust in different scenarios, since WiFi signals are very sensitive to different factors. CSI is impacted by not only human activities but also the static and motion status of WiFi transmitters, receivers, and the surrounding environment. For example, the location and orientation of WiFi receivers and target persons have a great impact on how CSI amplitude and phase change. When a recognition algorithm is trained or modeled by CSI measurements from a certain WiFi receiver, it is challenging to make the algorithm still work for another WiFi receiver placed at a different location with different antenna orientations. Moreover, different persons may have different motion and activity patterns, so models trained on one person may not work for another person whose data are not seen during training or modeling.

As shown in Fig. 4.1, CSI patterns of the same activity are very different for different persons or different receiver locations/orientations. Modeling-based and instance-based learning algorithms do not work if they are tested with unseen persons or unknown receiver locations/orientations. For example, when person 1 changes the status from standing to sitting, the CSI amplitude of receiver 1 decreases from $\sim 25\text{dB}$ to $\sim 10\text{dB}$. But for receiver 2, the CSI amplitude changes from $\sim 22\text{dB}$ to $\sim 20\text{dB}$. Moreover, there are no big differences for different activities for receiver 2 of person 2. Therefore, traditional

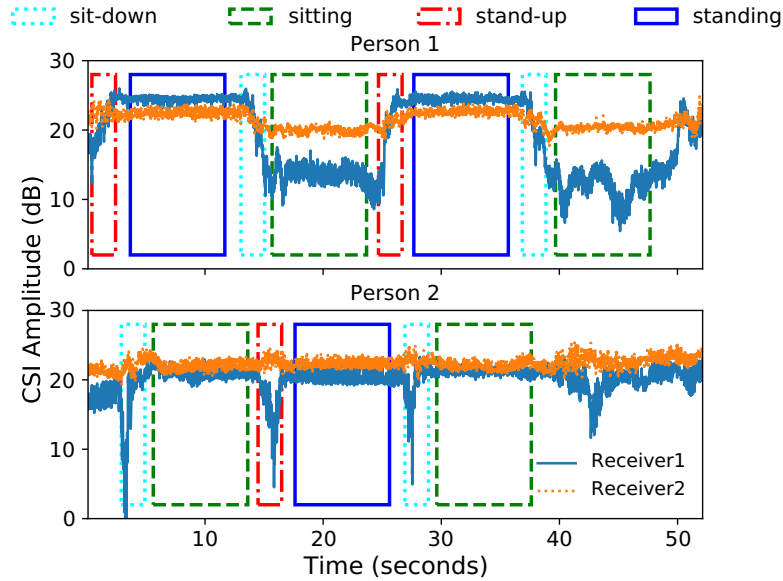


Figure 4.1: CSI amplitude of TX0/RX0 of the 1st subcarrier from two WiFi devices for different activities performed by two persons. Different persons or receivers have different CSI patterns. It is challenging to distinguish different activities if the recognition algorithm is trained or modeled with CSI data of receiver 1 for person 1 and tested with CSI data of receiver 2 for person 2.

recognition algorithms can hardly work if they are trained or modeled with CSI data of receiver 1 for person 1 and tested with CSI data of receiver 2 for person 2.

WiFi-based activity recognition can reuse Convolutional Neural Networks (CNNs) that have high performance for computer vision tasks. But reusing existing CNNs may result in low performance for WiFi-based activity recognition, since CSI has some unique characteristics that are different from images. CSI has much smaller spatial resolutions than images and contains noises and interferences from all directions. CSI amplitude and phase are very sensitive to the surrounding environment and the location and orientation of WiFi receivers and target persons. Therefore, pre-trained CNNs have low accuracy for unseen persons or unknown receiver locations and orientations. As shown in Fig. 4.2, depthwise separable 2D convolutions, or SeparableConv2D, achieves 99% accuracy when the data of testing persons and receivers are seen during training. But the accuracy drops to 84% for unseen persons and 62% for unseen persons and unknown receiver locations and orienta-

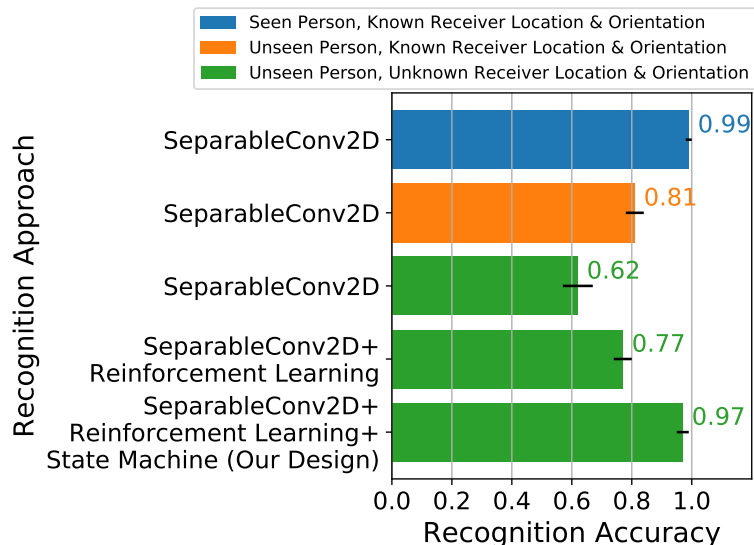


Figure 4.2: Accuracy comparison of different deep learning solutions for WiFi-based activity recognition. The recognition accuracy of SeparableConv2D drops dramatically for unseen persons or unknown receiver locations & orientations. The accuracy is significantly improved by the proposed design with reinforcement learning and state machine.

tions. Therefore, it is necessary to find the suitable CNN types, neural architectures and learning parameters that are specially designed for CSI data.

In this chapter, we propose a novel deep learning solution for robust WiFi-based activity recognition. The proposed design contains three neural networks: a 2D CNN as the recognition algorithm, a 1D CNN as the state machine, and a Recurrent Neural Network (RNN) with Long Short-Term Memory (LSTM) as the reinforcement learning agent for neural architecture search. In summary, the proposed design has the following three components.

Recognition Algorithm: 2D CNN. To learn location and person independent features from different perspectives of 4D CSI tensors in time, spatial, and frequency domains.

State Machine: 1D CNN. To learn temporal dependency information from previous classification results for improving the recognition performance of static and transi-

tion activities.

Neural Architecture Search: RNN with LSTM. To optimize the neural architecture of the recognition algorithm by reinforcement learning.

The combination of these three deep learning components provides location and person independent WiFi-based activity recognition with the following properties.

Robust: It is independent of the locations, placements, and orientations of WiFi devices and target persons. The pre-trained model also works when WiFi receivers are placed at unknown places with uncertain orientations and antenna placements and for new persons whose data are not seen during training.

Automatic: It requires very little human efforts for data collection, ground truth labeling, and training. It only needs simple CSI pre-processing and does not require manual efforts for ground truth labeling, feature engineering, signal processing, learning parameters tuning, or neural architecture search.

Reusable and adaptable: It can be trained on-the-fly on additional data and pre-trained models without restarting the training process from scratch. It can evolve over time as there are more data measured in new scenarios with different settings.

The proposed design is evaluated by CSI measurements from real-world scenarios. In total, there are 14555 instances of 5 activities, including sitting, standing, sit-down, stand-up, and walking, performed by 7 persons. There are 4 WiFi receivers placed at different locations with different antenna orientations. Each participant can sit or stand at two locations with random facing directions and walk randomly in a constrained area. As shown in Fig. 4.2, with reinforcement learning, the recognition accuracy is improved from 62% to 77% for unknown receiver locations and orientations and for unseen persons. The accuracy is further improved to 97% by adding both reinforcement learning and state machine. The proposed design is also evaluated by two public datasets, S.Yousefi-

2017 [205] and FallDeFi [110], with accuracy of 80% and 83%. In summary, we make the following contributions.

- We propose a novel deep learning solution for robust WiFi-based activity recognition. The proposed design uses a 2D CNN as the recognition algorithm, a 1D CNN as the state machine, and an RNN with LSTM for neural architecture search with reinforcement learning.
- The propose design recognizes 5 activities with 97% average accuracy when the location and orientation of WiFi receivers and target persons are unknown and the data of target persons are not seen during training.
- The propose design requires very little human efforts for ground truth labeling, signal processing, and model training. It can be easily re-trained on new data to evolve over time and be adaptive to different scenarios.

The rest of the chapter is organized as follows. §4.2 presents related works of on WiFi-based motion detection and activity recognition. §4.3 presents the proposed design. §4.4 shows the experiment setup and evaluation results. Section 4.5 presents some discussions on overhead of neural architecture search and robustness of new environments. §4.6 presents the summary of the chapter.

4.2 Related Work

CSI captures how wireless signals propagate from the transmitter to the receiver through the surrounding environment. CSI amplitude and phase are impacted by nearby persons doing different motions or activities. Recently, CSI is widely used for human motion detection and activity recognition. Readers may refer to the survey [94] for more details of signal processing techniques, algorithms, applications, challenges, and future trends of CSI-based sensing.

Table 4.1: Related works of fall and motion detection with CSI.

Reference	Signal Processing	Algorithm	Performance
WiFall [52]	Weighted Moving Average (WMA), Local Outlier Filter (LOF)	kNN, One-Class SVM	Fall Detection Precision: 87%
FallDeFi [110]	Wavelet Filter, DWT, STFT, PCA, Interpolation, Thresholding	One-Class SVM	Fall Detection Accuracy: 93%/80% (same/different environments)
RT-Fall [157]	STFT, Band-Pass Filter (BPF), Interpolation, Thresholding	One-Class SVM	True Positive Rate: 91%, True Negative Rate: 92%
Anti-Fall [214]	Interpolation, Low-Pass Filter (LPF), Threshold-Based Sliding Window	One-Class SVM	Precision: 89%, False Alarm Rate: 13%
WiSpeed [216]	Median Filter, ℓ_1 Trend Filter, Thresholding	Statistical Modeling, Peak Detection	Fall Detection Rate: 95%
WiKey [4, 5]	LPF, PCA, DWT	kNN+DTW	Keystroke Detection: 97.5%
MAIS [30]	LPF, Outlier Filter, Thresholding	kNN	Anomaly Detection: 98.04%
FRID [38]	N/A	CSI Phase Coefficients	Motion Detection Precision: 90%
MoSense [42]	LPF, Euclidean Distance, Thresholding	Binary Classification	Motion Detection Accuracy: 97.38%/93.33% (LoS/NLoS, 5 activities)
AR-Alarm [81]	Interpolation, BPF, Duration-Based Filter	Binary Classification	True Positive Rate: 98.1%/97.7%
Liu-2017 [87]	Signal Isolation by Skewness	One-Class SVM	Motion Detection Rate: 90.89%
Wi-Sleep [89, 90]	Hampel Filter, Wavelet Filter, DWT, Interpolation	Pattern Matching	Posture Change Detection: 83.3%
SEID [91]	Signal Compression by CSI Amplitude Variance	HMM	Motion Detection Precision: 98%
WiStep [202]	Long Delay Removal, DWT, BPF, PCA	Peak Detection, Threshold-Based Detection	True Positive Rate: 96.41%, False Positive Rate: 1.38%
NotiFi [232]	PCA	Dynamic Hierarchical Dirichlet Process, Bayesian Nonparametric Model	Abnormal Activity Detection Accuracy: 89.2%/ 85.6%/75.3% (LoS/NLoS/through-wall)

4.2.1 Motion Detection with CSI

In recent years, CSI is widely used for fall detection [157, 110, 52, 214, 216, 147] and motion detection [42, 87, 38, 81, 30, 202, 89, 90, 5, 232, 91]. Table 4.1 shows a summary of the signal processing techniques, algorithms, and performance results of CSI-based fall and motion detection. Motion detection is a relatively simple task and sometimes has no clear borderline between signal processing and the detection algorithm. After some signal processing techniques such as low-pass filters and thresholding, the detection result can be directly derived without further detection or classification algorithms. Modeling-based algorithms, e.g., threshold-based detection and peak detection, and very simple learning-based algorithms, e.g., one-class Support Vector Machine (SVM), are widely used for WiFi-based motion detection. Theoretical and statistical models are usually very sensitive to noises and outliers, so noise reduction is usually needed, such as the Hampel filter, wavelet filter, and local outlier filter. Aryokee [147] also uses CNNs and state machine, but its objective is fall detection with radar signals while ours is activity recognition with WiFi signals. Radar is designed for sensing and has finer granularity than WiFi which is designed for communication but not for sensing. So activity recognition with WiFi is much harder than fall detection with radar. Our goal is to not only detect motions but also recognize different motion and static activities with high and robust performance. The proposed design can distinguish different static and motion activities with 97% average accuracy for leave-one-person-out and leave-one-device-out validation. Besides, Aryokee uses HMM as the state machine which needs to be trained separately and has low performance for CSI data that involve non-linearities and long-term temporal dependencies. Our design has the recognition algorithm and state machine trained together and the neural architecture automatically optimized by reinforcement learning.

Table 4.2: Related works of activity recognition with CSI.

Reference	Signal Processing	Algorithm	Accuracy
Wi-Chase [6]	LPF	kNN, SVM	97% (3 activities)
WIBECAM [23]	N/A	Autoregressive Model	73% to 100% (4 activities)
BodyScan [27]	LPF, PCA, Thresholding	SVM	72.3% (5 activities)
MAIS [30]	LPF, Outlier Filter, Thresholding	kNN	93.12% (3 activities)
DFLAR [33]	N/A	Sparse Auto-Encoder	90% (8 activities)
HuAc [43]	Outlier Filter, WMA; LPF, Thresholding, k Means	SVM	93% (16 activities)
EI [66]	Hampel Filter; Thresholding	CNN	<75% (10 users, 6 activities)
Wang-2018 [158]	Median Filter, Linear Fitting, LPF	SOM, Softmax Regression	>85% (8 activities)
CARM [165, 166]	DWT, Thresholding, PCA	HMM	>96% (8 activities)
Wang-2015 [173]	Gaussian Filter, LOF, k Means	DTW, SVM	80% (13 activities)
E-eyes [170]	LPF, Thresholding, Clustering	Multi-Dimensional DTW, Pattern Matching	90%/95% (single device/multiple devices, 13 activities)
Wei-2015 [176]	Exponential Smoothing	Sparse Representation	<90% (8 activities)
ARM [189]	Wavelet Filter; DWT	DTW, HMM	>75% (6 activities)
Zeng-2015 [211]	BPF	Decision Tree, Simple Logistic Regression	89.6%/94.75 (entrance/in store, 4 activities)
WiDriver [25]	Signal Compression by Neural Network	Fresnel Zone Model, Finite Automata	96.8% (11 postures), 90.76% (7 activities)
HeadScan [29]	LPF, PCA	Sparse Representation, ℓ_1 Minimization	86.3% (5 activities)
WiBot [125]	LPF, PCA	kNN	94.5%/90.5% (3/5 activities)
SEARE [192]	LPF, Median Filter, PCA, Thresholding	DTW	97.8%/91.2% (LoS/NLoS, 4 activities)

4.2.2 Activity Recognition with CSI

CSI is commonly used for recognizing human activities, including daily activities [6, 23, 27, 30, 33, 43, 66, 158, 165, 166, 173, 170, 176, 189], shopping [211], driving [25, 125], exercising [192], and head & mouth activities [29]. Table 4.2 shows a summary of the signal processing techniques, algorithms, and recognition accuracy of CSI-based activity recognition. Almost all the recognition applications use learning-based algorithms as the classifier. SVM and k Nearest Neighbor (kNN) are two of the most used classifiers for CSI-based activity recognition. Dynamic Time Wrapping (DTW) is usually used for kNN as the distance metric. Learning-based algorithms are usually not very sensitive to noises and outliers. Many learning-based algorithms use none or very simple noise reduction methods such as averaging and median filter. Noise reduction is usually used for modeling-based algorithms which are typically sensitive to noises. The major issue for modeling-based and instance-based learning algorithms is that they are not location or person independent when the data of testing devices or persons are not seen during training or modeling. Another issue for instance-based learning algorithms is that they need to calculate the distance from the testing instance to all the training instances. This introduces expensive overhead when there are multiple classes and each class instance has many CSI data points. SVM, kNN, and DTW have high inference costs for calculating the distance of different samples, so they usually employ feature extraction, subcarrier selection, or dimension reduction to reduce the input size. EI [66] uses a CNN as the recognition algorithm, but its recognition accuracy is less than 75%. SignFi [95] also uses a CNN for gesture recognition with WiFi, but it is tested with known WiFi receiver locations and orientations and has low accuracy for leave-one-person-out tests. Since CSI data are different from images and videos, it may result in low recognition performance by just reusing CNNs that are designed for computer vision tasks. It is necessary to find the suitable CNNs, including the CNN type, neural architecture, and neural weights, that are specifically designed for CSI data. To address this issue, we use different convolutions

as the recognition algorithm for learning location and person independent features from different perspectives of CSI data. Moreover, we use reinforcement learning for optimizing the neural architecture of the recognition algorithm and a lightweight 1D CNN as the state machine for learning temporal dependencies. The proposed deep learning design has 97% average accuracy for leave-one-person-out and leave-one-device-out validation.

4.3 Design

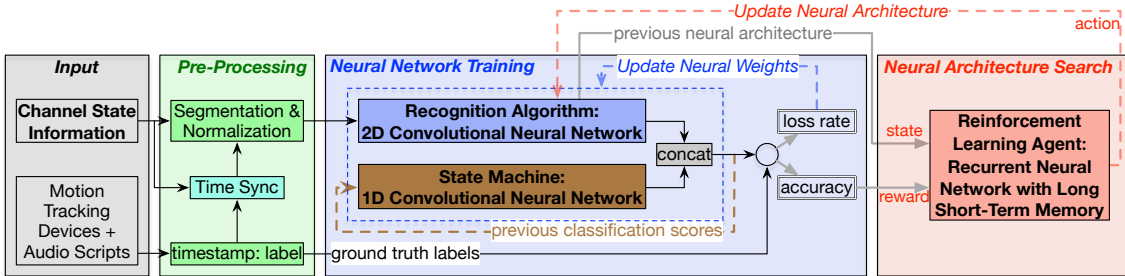


Figure 4.3: The training process of the proposed design. The input is a time series of CSI matrices measured by WiFi packets. The recognition algorithm is a 2D CNN. The state machine is a 1D CNN. The final classification results are calculated by the concatenation of the recognition algorithm and the state machine. The neural architecture of the recognition algorithm is updated by the reinforcement learning agent by an RNN with LSTM. Motion tracking devices and audio signals are used for ground truth labeling during off-line training. During the inference stage, only CSIs are used as the input.

This section presents the details of the proposed design, including pre-processing, recognition algorithm, state machine, and neural architecture search. The overview of the proposed design is shown in Fig. 4.3. During data collection, each participant follows the audio scripts to perform different activities. At the same time, CSI measurements are collected by WiFi receivers. A motion tracking system, HTC Vive, along with the audio scripts, are used to label the CSI data. Note that motion tracking devices and audio scripts are used only for off-line training. During the inference stage, only CSI measurements are used as the input. The proposed design has the following components. First, a time series of CSI matrices is synchronized and segmented by ground truth labels and then normalized by the training data. Second, the recognition algorithm uses a 2D or 3D CNN

to learn features from different perspectives of 4D CSI tensors in time, spatial, and frequency domains. Third, the state machine uses a 1D CNN to learn temporal dependency information from previous classification results. Finally, the reinforcement learning agent uses an RNN with LSTM to optimize the neural architecture of the recognition algorithm.

The combination of these components provides location and person independent WiFi-based activity recognition. The recognition algorithm is responsible for learning location and person independent features from different domains within one CSI segment. The state machine tries to learn temporal dependencies across multiple CSI segments. The reinforcement learning agent optimizes the neural architecture of the recognition algorithm to maximize the accuracy. As a result, the proposed design is robust in new scenarios when the locations and orientations of WiFi devices and target persons are unknown and for new target persons whose data are unseen during training. It requires very little human efforts for ground truth labeling, signal processing, feature engineering, parameter tuning, and neural architecture search.

4.3.1 Pre-Processing: Normalization of CSI Amplitude

CSI represents how wireless signals travel from the transmitter to the receiver at certain carrier frequencies along multiple paths. For a MIMO-OFDM channel with N_t transmit antennas, N_r receive antennas, and N_c subcarriers, the CSI is a 3D matrix $H \in \mathbb{C}^{N_r \times N_t \times N_c}$. Each CSI entry is a complex number representing the Channel Frequency Response (CFR) of the multi-path channel:

$$h(f; t) = \sum_{i=1}^N a_i(t) e^{-j2\pi f \tau_i(t)}, \quad (4.1)$$

where $a_i(t)$ and $\tau_i(t)$ are the power attenuation and propagation delay, respectively, of the i -th path, f is the carrier frequency, and N is the number of multi-path components [148]. The CSI amplitude and phase represent the power attenuation and phase shift of the multi-path channel, which are impacted by the multi-path effects and the static/mobility

status of the transmitter, receiver, and nearby humans/objects. CSI phase is too sensitive to very small environmental changes, so we only use CSI amplitude as the input.

During data collection, CSI measurements are collected from multiple WiFi receivers every 10 milliseconds and are synchronized with the timestamps of audio scripts. During off-line training, raw CSI measurements are segmented based on the corresponding ground truth labels from the audio scripts. Each CSI segment has time duration of 2 seconds containing 200 samples of CSI matrices. Shorter CSI segments are discarded. There are $N_t = 3$ transmit antennas, $N_r = 3$ receive antennas, and $N_c = 30$ subcarriers in our experiments, so the size of each CSI matrix is $(3, 3, 30)$. Each training and testing sample is a time series of 3D CSI matrices, resulting in a 4D CSI tensor with size of $(200, 3, 3, 30)$. Each training and testing CSI segment is normalized by the CSI amplitude of training segments:

$$x_{train}^i = \frac{|csi_{train}^i| - \text{mean}(|csi_{train}|)}{\text{std}(|csi_{train}|)}, \quad i = 1, \dots, N_{train}; \quad (4.2)$$

$$x_{test}^j = \frac{|csi_{test}^j| - \text{mean}(|csi_{train}|)}{\text{std}(|csi_{train}|)}, \quad j = 1, \dots, N_{test}, \quad (4.3)$$

where $\text{mean}(|csi_{train}|)$ and $\text{std}(|csi_{train}|)$ are the mean and standard deviation of the CSI amplitude of training samples. Each dimension of the 4D CSI tensor is normalized separately. Note that $\text{mean}(|csi_{train}|)$ and $\text{std}(|csi_{train}|)$ do not include any testing CSI samples, so the information of testing CSI samples is not leaked to the normalized training data x_{train}^i .

We use as little pre-processing as possible to retain as much raw information as possible. CSI amplitude normalization is the only pre-processing before feeding the input to the recognition algorithm. We leverage the power of DNNs and reinforcement learning for extracting useful information from the normalized CSI input. When there are more pre-processing involved, there will be a higher probability of losing information embodied in the unprocessed data. Moreover, the CSI pre-processing has very low computation overhead, so it runs fast for both off-line training and real-time inference. Fig. 4.4 shows an example of the pre-processed CSI segment. It contains CSI amplitude variations in

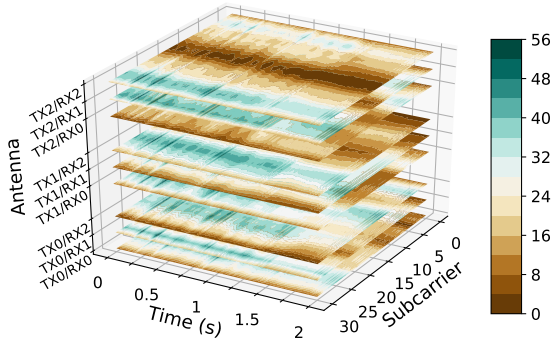


Figure 4.4: A time series of CSI amplitude measurements provides information in time, spatial, and frequency domains. The CSI tensors are fed to different CNNs to automatically find location and person independent features.

the time, spatial, and frequency domains. The pre-processed 4D CSI tensors are fed to different CNNs to learn useful features from different perspectives of the CSI data for location and person independent activity recognition.

4.3.2 Recognition Algorithm: 2D/3D CNN

CSI matrices have some similar attributes as digital images, so WiFi-based activity recognition can reuse the algorithms that have high performance for computer vision tasks. For a MIMO-OFDM channel with N_t transmit antennas, N_r receive antennas, and N_s subcarriers, the CSI matrix is similar to a digital image with spatial resolution of $N_t \times N_r$ and N_s color channels. Convolutional Neural Networks (CNNs) are proven to have very good performance and are used in almost all modern neural network architectures. Therefore, WiFi-based activity recognition can reuse the CNN models and architectures designed for computer vision tasks.

However, CSI has some unique characteristics that are different from images and videos, so reusing existing CNNs may result in low performance for WiFi-based activity recognition. For example, the spatial resolution, which is 3×3 in our case, is much smaller than that of images. A digital image usually have 3 (RGB) or 1 (grayscale) color channels, while an uncompressed CSI matrix has 52 subcarriers, including 48 data subcarriers and 4 pilot subcarriers, for a 20MHz WiFi channel. Besides, unlike images and videos

that usually contain light signals from visible directions and distances, CSI may contain noises and interferences from all directions. So CSI amplitude and phase are very sensitive to the surrounding environment and the location and orientation of WiFi receivers and target persons. Therefore, we need to find which types of DNNs are suitable for CSI data.

A time series of CSI matrices characterizes MIMO channel variations in different domains, i.e., time, frequency, spatial, as shown in Fig. 4.4. CSI can be processed, modeled, and trained in different domains for different WiFi sensing applications. Different CNNs can infer information from their specific aspects of the training data. Our task is to find the best type of CNNs and the corresponding neural architecture that provide robust WiFi-based activity recognition for unknown receiver locations/orientations and unseen persons. We consider the following three types of convolutions: 2D convolutions, 3D convolutions, and depthwise separable 2D convolutions.

2D Convolution (Conv2D) is calculated by

$$S(i, j) = (I * K)(i, j) = \sum_x \sum_y I(x, y)K(i - x, j - y), \quad (4.4)$$

where S is the convolution output, I is the input, and K is the kernel [40]. The convolutional layer learns the weights and biases while going through the input vertically and horizontally with the same kernel. For Conv2D, the same kernel is shared for different color channels.

3D Convolution (Conv3D) uses 3D kernels, instead of 2D kernels in Conv2D, as going through cubic regions of the input data. Conv3D is calculated by

$$S(i, j, d) = \sum_x \sum_y \sum_z I(x, y, z)K(i - x, j - y, d - z). \quad (4.5)$$

Compared with Conv2D, Conv3D learns features combined in all three domains: time, spatial, and frequency. Besides, CSI tensor reshaping is not needed for Conv3D, while it is necessary for 2D convolutions. One potential issue for using Conv3D for CSI data is

that CSI matrices have too small spatial resolutions, i.e., 3×3 in our case. To address this issue, we reshape the CSI tensors from $\mathbb{R}^{N_s \times N_r \times N_t \times N_c}$ to $\mathbb{R}^{N_s \times N_c \times N_t \times N_r}$, with N_s as the number of CSI samples for each segment. Another issue of Conv3D is that it has much higher computation overhead than Conv2D.

Depthwise Separable 2D Convolution (SeparableConv2D) first uses different kernels (depthwise convolutions) for each color channel and then uses a 1×1 kernel (pointwise convolutions) along the input depth to get the combined features. SeparableConv2D is calculated by:

$$D(i, j, d) = \sum_x \sum_y I(x, y) K_d(i - x, j - y), \quad S(i, j, d) = \sum_d D(i, j, d) K_p(k - d), \quad (4.6)$$

where $D(i, j, d)$ is the output of the first step, K_d is the kernel of the d -th color channel, K_p is a 1×1 pointwise convolution kernel [18, 10], and $S(i, j, d)$ is the convolution output of the second step. For computer vision tasks, SeparableConv2D has about 10 times less computation cost with a small reduction of accuracy compared with normal convolutions [18, 10]. For CSI data, SeparableConv2D has the best recognition accuracy, as shown in Section 4.4.

DNNs are organized into multiple layers, and the convolutional layer is only one of the layers. Each convolutional layer is usually followed by other layers such as batch normalization, ReLU, max-pooling, and dropout. Before the output layer, a flatten layer and a softmax layer are needed to calculate the loss rate of the classification algorithm. CNNs learn the training parameters of each layer, using an optimization algorithm, to minimize the loss rate. Since the convolution layer shares the same kernel for multiple input regions, it significantly reduces computation overhead for both training and inference.

4.3.3 State Machine: 1D CNN

There is temporal dependency information within a single CSI segment and across multiple CSI segments. Each CSI segment is a 4D tensor containing 200 samples of CSI

matrices measured in 2 seconds. The temporal dependencies within each CSI instance are learned by the recognition algorithm. There are also temporal dependencies among neighboring CSI segments that are not learned by the recognition algorithm. For example, if the classification result of the current CSI segment is stand-up, the next CSI segment has a high probability to be classified as standing. Therefore, we add a state machine to learn the temporal dependencies across CSI segments for improving the recognition accuracy. Fig. 4.5 shows the state transition diagram of 5 human activities. Each state is in corresponding to a 4D CSI tensor with size of (200, 3, 3, 30). The state machine is used to learn the state transition probabilities and to predict the current state based on previous states. The final classification results are obtained by the concatenation of the output of the state machine and the classification scores of the recognition algorithm.

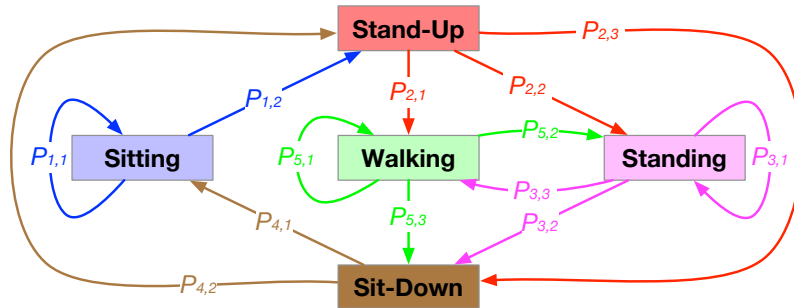


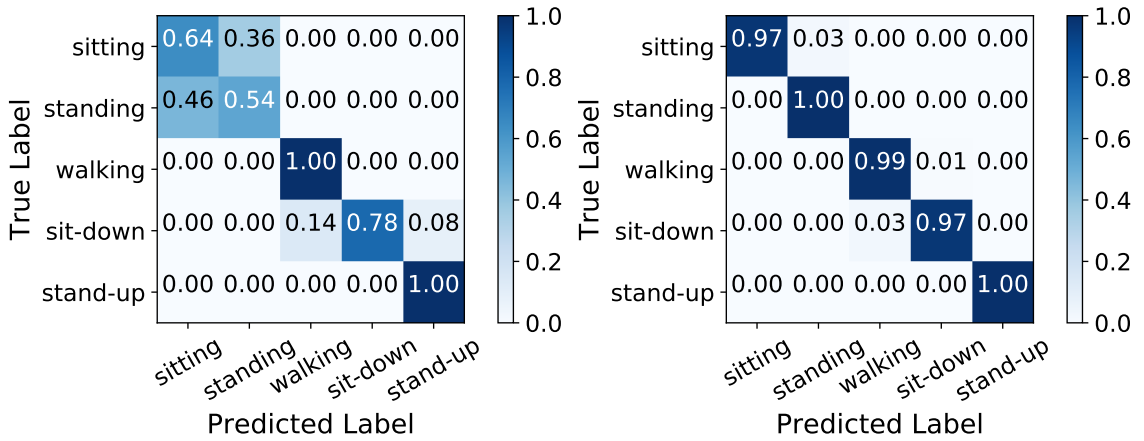
Figure 4.5: State machine of 5 activities in the experiments.

The state machine can be modeled by a Markov chain which represents a time series of possible activities. The probability of each activity depends on the state of the previous activity. Hidden Markov Model (HMM) is a widely used Markov model wherein the states are modeled by a Markov process with unobservable states, i.e., hidden states. HMM has a strong assumption that state transitions only depend on the current state which can be modeled by a linear transformation of the previous state. This assumption does not hold for the CSI data wherein state transitions have non-linear relationships with the current and previous states. Besides, HMM needs parameter learning to find the best set of state transitions and emission probabilities. The learned parameters of HMM are highly

dependent on the training data. When the state transitions of training and testing data have different distributions, the learned HMM will overfit to training data and give low accuracy for testing data. Moreover, the HMM needs to be trained separately in addition to the training of the recognition algorithm.

Recently, DNNs, especially RNNs, are popular replacements of HMM for sequential problems. RNNs provide good performance for complex sequential inputs that involve non-linearities and long-term temporal dependencies, which are hard to handle for HMM. DNNs do not have the Markov assumption. Instead, DNNs rely on the learning parameters, or neural weights, to extract complex features and state transitions. RNNs are good at capturing non-linearities and long-term temporal dependencies and are proven to have good performance for sequential inputs. However, RNNs have extremely high computation costs. Besides, we run experiments and find that RNNs have much lower accuracy than CNNs for CSI data. The major reason is the low spatial resolution of a single CSI matrix. It is hard for RNNs to capture short-term temporal dependencies within a CSI segment. We use a 1D CNN, i.e., Conv1D, as the state machine. 1D CNNs run much faster than RNNs and offer comparable or higher performance for CSI data. It has only one Conv1D layer with 5 kernels of size 1×1 . The state machine is very lightweight with only 140 parameters, including 10 from the Conv1D layer and 130 from the softmax layer. It offers 20% accuracy improvements with very low computation and training costs, which is shown in Section 4.4. Moreover, the state machine is trained together with the recognition algorithm, so it does not require extra efforts to train the recognition algorithm and state machine separately.

Fig. 4.6 shows the confusion matrices of SeparableConv2D with and without the state machine. Both confusion matrices are obtained from leave-one-person-out and leave-one-device-out validation for person 2. Without the state machine, the recognition algorithm has difficulties in distinguishing sitting and standing from each other. The reason is that CSI measurements have similar patterns for static activities. This issue is addressed by the state machine, which improves the overall recognition accuracy from 78% to 99%. As



(a) SeparableConv2D with reinforcement learning, without state machine. Accuracy = 78%. (b) SeparableConv2D with reinforcement learning and state machine. Accuracy = 99%.

Figure 4.6: Confusion matrix of leave-one-person-out and leave-one-device-out testing results. The recognition performance for static activities, i.e., sitting and standing, is significantly improved by the state machine.

shown in Fig. 4.6b, the recognition performance of sitting and standing has significant improvements. This is accomplished by utilizing the high accuracy of motion transition activities and the temporal dependencies learned by the state machine. More details of the impact of the state machine are shown in Section 4.4.

4.3.4 Neural Architecture Search: Reinforcement Learning

Although the features, or learning weights, of CNNs can be automatically learned during training, it is non-trivial to find the best neural architecture, especially for CSI data. The neural architecture of a CNN refers to a set of hyperparameters including the number of convolutional layers, number of convolutional kernels, size of convolutional kernels, size of max-pooling, and dropout rate. One way is to reuse the neural architectures that are trained by a lot of data and proven to provide high performance for computer vision and natural language tasks. But these neural architectures do not necessarily give good performance for WiFi-based activity recognition, since CSI data are different from images, videos, and texts. Another approach is using neural architecture search which tries to optimize the CNN architecture for improving the classification performance.

We use a reinforcement learning agent for neural architecture search [237]. It needs almost no human efforts for hyperparameters tuning. The reinforcement learning agent tries to maximize a numerical reward signal by learning how to interact with the environment in discrete time steps [145]. In the context of neural architecture search, the environment is the recognition algorithm which updates the state and reward to the agent. The action signal is the neural architecture of the recognition algorithm, and the reward signal is the classification accuracy.

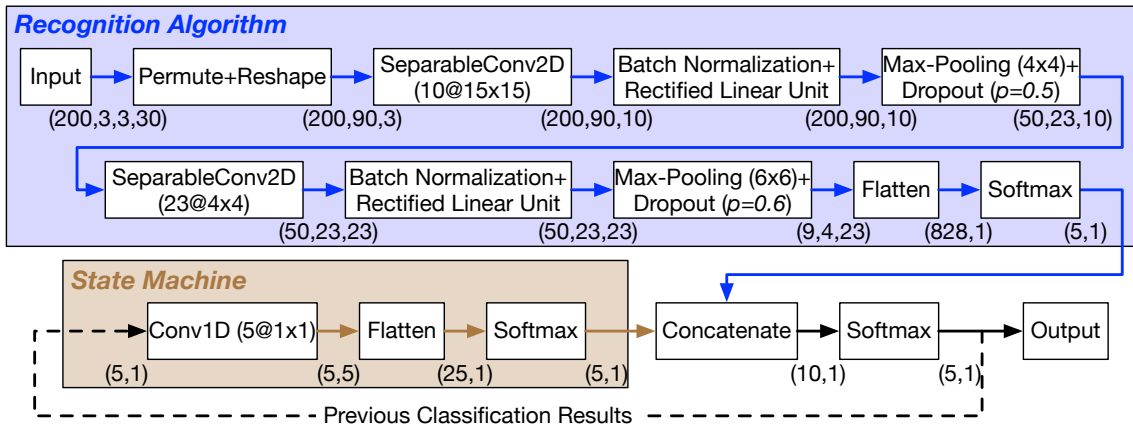


Figure 4.7: Neural architecture of the best performing recognition algorithm. The input is pre-processed 4D CSI tensors, and the output is one of the five activities. The numbers inside the brackets indicate the output size of each layer.

We use NASCell [237] from TensorFlow as the reinforcement learning agent. The recognition algorithm is selected from three types of convolutions: Conv2D, Conv3D, and SeparableConv2D. The reinforcement learning agent uses an RNN with LSTM to update the neural architecture of the recognition algorithm. For each training cycle, the training results and neural architecture of the recognition algorithm are fed to the reinforcement learning agent. The reinforcement learning agent uses the previous neural architecture as the state signal and the recognition accuracy as the reward signal to find the action output. The reinforcement learning agent updates actions to maximize the expected reward, which

is done by the policy gradient of the empirical approximation of the expected reward

$$\nabla_{\theta_c} J(\theta_c) \approx \frac{1}{m} \sum_{k=1}^m \sum_{t=1}^T \nabla_{\theta_c} \log P(a_t | a_{(t-1):1}; \theta_c) (R_k - b), \quad (4.7)$$

where $\nabla_{\theta_c} J(\theta_c)$ is the policy gradient of the expected reward $J(\theta_c)$ with learning parameters θ_c , m is the number of neural architectures that NASCell samples in one batch, T is the number of hyperparameters needed to design a neural architecture, a_t is the list of actions, R_k is the training accuracy of the k -th neural neural architecture, and b is the average training accuracy of previous neural architectures for preventing high variances [237]. The action output of NASCell is mapped to the neural architecture of the recognition algorithm to start the next training cycle. The recognition algorithm is implemented by Keras with Conv2D, Conv3D, and SeparableConv2D, and the reinforcement learning agent is implemented by TensorFlow with NASCell.

Fig. 4.7 shows the best performing neural architecture of the recognition algorithm and state machine optimized by the reinforcement learning agent. Note that the neural architecture of the state machine is fixed and is not updated by the reinforcement learning agent. The recognition algorithm has two SeparableConv2D layers with each followed by batch normalization, Rectified Linear Unit (ReLU), max-pooling, and dropout layers. The first SeparableConv2D layer has 11 kernels of size 22×22 , and the second one has 10 kernels of size 15×15 . The size of the two max-pooling layers is 3×3 and 6×6 , respectively. The dropout rates are 0.5 and 0.6. The input of the recognition algorithm is 4D CSI tensors of size (200, 3, 3, 30), and the output is the classification scores of 5 activities. The state machine contains a Conv1D layer with 5 kernels of size 1×1 . The input of the state machine is the classification scores of the previous CSI segment, and the output is the classification scores of 5 activities. The final classification output is calculated by the concatenation of the outputs of the recognition algorithm and state machine. The neural architecture of the recognition algorithm and state machine has 5534 trainable and 66 non-trainable parameters in total. Non-trainable parameters are the mean and variance

of batch normalization layers which are not trained with backpropagation.

4.4 Evaluation

Table 4.3: Overview of Performance Evaluation for Different Test Scenarios

	Dataset	Size (GB) ¹	Test Scenario	# (TXs, RXs, Rooms, Persons)	# Instances	Size of Each Instance	Test Accuracy (CNN, CNN+RL, CNN+RL+SM)
§4.4.1	This paper	17.05	Same environment; unseen persons; unknown receiver location/orientation	(1, 4, 1, 7)	14555	(200, 3, 3, 30)	62%, 77%, 97%
§4.4.2	S.Yousefi-2017 [205]	2.78	Same environment; unseen persons; known receiver location/orientation	(1, 1, 1, 6)	2079	(2000, 3, 30)	45%, 63%, 80%
	FallDeFi [110]	1.06	Unseen environment; unseen persons; unknown receiver location/orientation	(5, 5, 6, 5)	397	(2000, 3, 30)	51%, 64%, 83%

¹ Size of numpy array of CSIs of 5 activities. Other activities of S.Yousefi-2017 [205] and FallDeFi [110] are not included.

An overview of performance evaluation of different testing scenarios is shown in Table 4.3. The proposed design is evaluated by leave-one-person-out and leave-one-device-out tests with CSI measurements of 5 activities performed by 7 persons with different receiver locations and orientations in Section 4.4.1. We also evaluate the proposed design with other public datasets including FallDeFi [110] and S.Yousefi-2017 [205] in Section 4.4.2.

4.4.1 Evaluation Results of Unseen Persons and Unknown Receiver Locations/Orientations

This section presents evaluation results of the impact of convolution types, state machine, and reinforcement learning for unseen persons and unknown receiver locations/orientations.

Experiment Setup and Data Collection

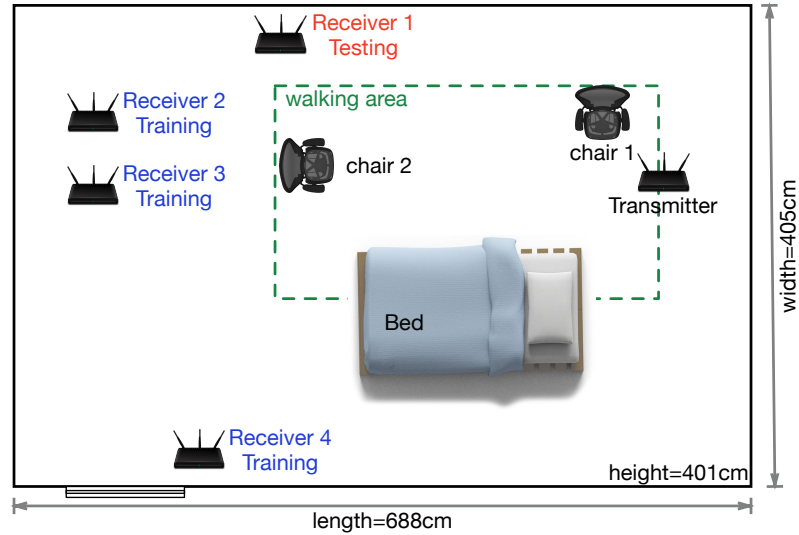


Figure 4.8: Experiment setup. CSI measurements of receiver 1 are for testing and other receivers for training. There are 4 WiFi receivers placed at different locations with different heights and antenna orientations. Each participant can walk randomly within the walking area, sit or stand at two chair locations with random facing directions.

The experiment setup is shown in Fig. 4.8. There are 4 WiFi receivers placed at different locations with different antenna orientations. CSI measurements of 3 receivers are used for off-line training and the other receiver is used for testing. There are 5 activities, sitting, standing, sit-down, stand-up, and walking, performed by 7 persons. The 7 participants have a wide variety of heights (from 64 to 75 inches) and weights (from 160 to 210 pounds), as shown in Table 4.4. In total, there are 14555 CSI segments, each with size of (200, 3, 3, 30), measured from 4 WiFi receivers. WiFi receivers are placed at different places with different heights and antenna orientations. During data collection, each person follows the audio instructions to perform different activities. Each person can walk randomly in the walking area and sit/stand at two locations with different facing directions. All activities are performed normally as in the real-life. For example, the person can have minor motions, such as interacting with the smartphone, as sitting, standing, and walking. CSI measurements are collected at different dates.

There are 4 WiFi receivers collecting CSI measurements as the participant is performing different activities following the audio instructions. Before each round of CSI

Table 4.4: Height and weight of experiments participants.

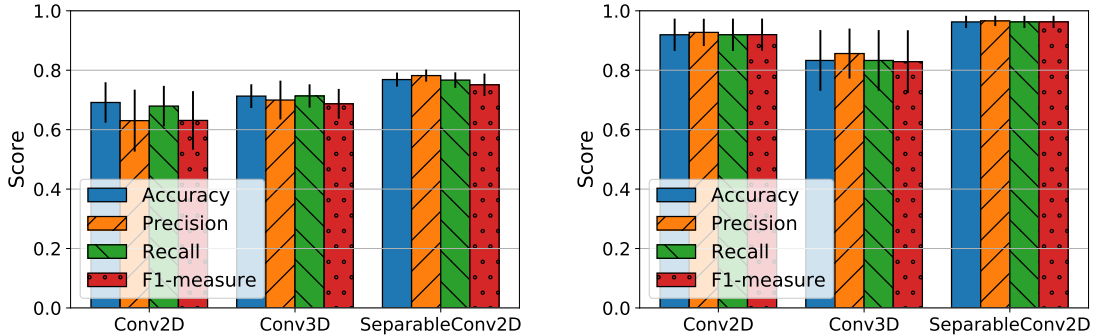
Height (inches)	67	75	64	70	75	66	66
Weight (pounds)	190	200	160	180	185	195	210

measurements, the timestamps of audio instructions and the 802.11n CSI tool of multiple devices are synchronized by the Network Time Protocol (NTP) with millisecond-level accuracy. The motion tracking devices and audio instructions are used later for ground truth labeling and segmentation of CSI measurements. Each WiFi device is a HummingBoard Edge [139] with an Intel 5300 WiFi card installed. The 802.11n CSI tool [51] is used for sending WiFi packets and measuring CSIs every 10 milliseconds. There are three antennas for each WiFi device, and the antenna spacing of each WiFi device is 2.6cm.

Raw CSI measurements are fed to Python scripts for extracting CSI matrices and pre-processing including segmentation and normalization. Both CNNs and NASCell use the Adam optimizer with learning rate of 0.01 and 0.001, respectively. The off-line network training is performed on a GTX 1080 Ti GPU. The performance is evaluated with leave-one-person-out tests, i.e., the data of testing persons are not seen during training, and leave-one-device-out tests, i.e., the location and orientation of testing receivers are unknown. The following performance results are evaluated with both leave-one-person-out and leave-one-device-out tests, i.e., neither testing devices nor persons are seen during training, unless stated otherwise. Performance metrics include accuracy, recall, precision, and F1-measure scores [133].

Impact of Different Convolutions

Fig. 4.9 shows the performance results of Conv2D, Conv3D, and SeparableConv2D with and without the state machine. Both leave-one-person-out and leave-one-device-out validation are used, i.e., CSI samples of the testing persons and testing devices are not seen during training. SeparableConv2D provides the best recognition performance. First, the average score, including accuracy, precision, recall, and F1-measure, of SeparableConv2D



(a) without state machine, with reinforcement learning (b) with state machine, with reinforcement learning

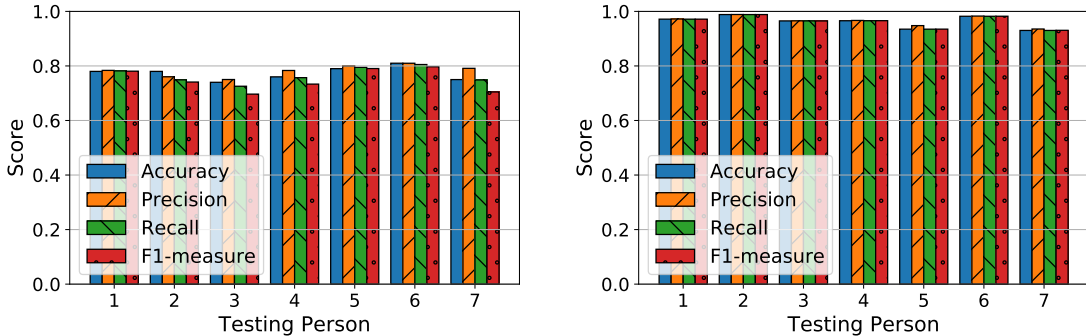
Figure 4.9: Average testing results of different convolution types for leave-one-person-out and leave-one-device-out validation. SeparableConv2D has the highest average score and the smallest standard deviation of recognition performance.

is the highest for both with and without the state machine. This means SeparableConv2D gives the most accurate recognition results. Second, the stand deviation of SeparableConv2D is the smallest, which means it gives consistent recognition results for different persons. Conv2D shares the same kernel for different color channels, so it does not learn features in the depth axis. Conv3D uses a 3D kernel to go through the CSI data and learn features in time, spatial, and frequency domains. But the CSI tensor has very small spatial resolution, i.e., 3×3 , so it does not help much learning spatial features. Conv3D learns features of different domains simultaneously using one shared kernel, while SeparableConv2D uses different kernels for different color channels and learns features in the depth axis separately in different kernels. So SeparableConv2D has the best recognition performance for CSI-based activity recognition.

Impact of State Machine

Comparing Fig. 4.9a and 4.9b, the state machine has a significant impact on the recognition performance. The average recognition accuracy of SeparableConv2D is 77% without the state machine and 97% with the state machine. The major contribution of the state machine is on improving the recognition performance for static activities, i.e., sitting and standing, by taking advantage of the temporal dependencies of multiple CSI segments.

Motion activities have different CSI patterns, so they have relatively high recognition accuracy even without the state machine. The state machine learns temporal dependencies of neighboring CSI segments and utilizes the high accuracy of motion activities to improve the accuracy of static activities. The probability of transition activities being misclassified as walking is also decreased by the temporal dependencies learned by the state machine.

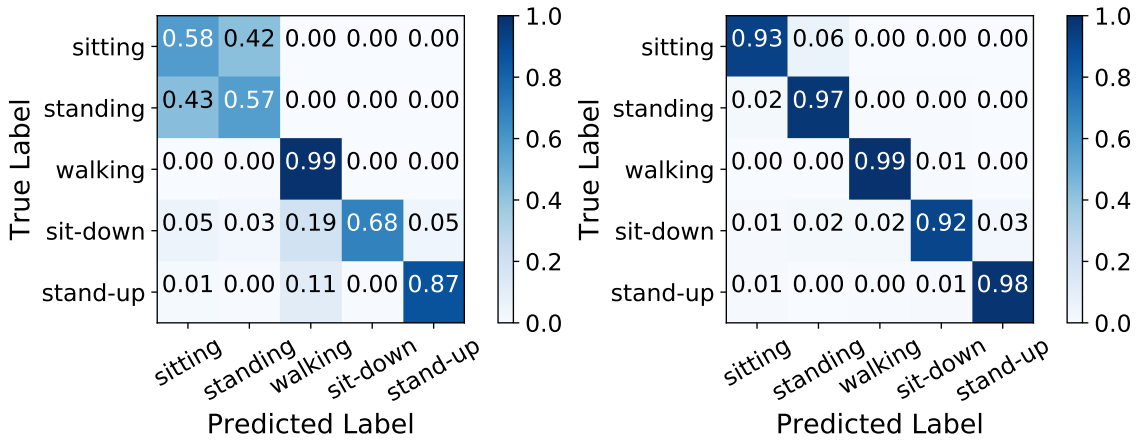


(a) without state machine, with reinforcement learning (b) with state machine, with reinforcement learning

Figure 4.10: Testing results of different persons for leave-one-person-out and leave-one-device-out validation.

Recognition scores of the best performing SeparableConv2D architecture with and without the state machine are shown in Fig. 4.10. Without the state machine, the recognition accuracy is 78%, 78%, 74%, 76%, 79%, 81%, and 75% for each testing person, as shown in Fig. 4.10a. With the state machine, the recognition accuracy is improved to 97%, 99%, 97%, 97%, 95%, 99%, and 93% for each testing person, as shown in Fig. 4.10b. The overall recognition accuracy is improved by 20% when the state machine and reinforcement agent are added. More details of why and how the state machine improves the recognition performance are shown in the following.

Fig. 4.11 shows the confusion matrices of the best performing SeparableConv2D architecture without the state machine. First, walking has close to 100% accuracy for all testing persons. Second, sitting and standing have very low recognition accuracy for all testing persons. Sitting and standing are very easy to be misclassified with each other



(a) SeparableConv2D with reinforcement learning, without state machine. Average recognition accuracy = 77%. (b) SeparableConv2D with reinforcement learning and state machine. Average recognition accuracy = 97%.

Figure 4.11: Confusion matrix of leave-one-person-out and leave-one-device-out testing results for all persons. The recognition performance for static activities, i.e., sitting and standing, is significantly improved by the state machine.

because they have similar CSI patterns. The only difference between them is the heights of sitting and standing. Some persons have no big differences between sitting and standing heights. Sitting and standing are static activities, so almost none of them are misclassified as motion activities. Finally, transition activities, i.e., sit-down and stand-up, have higher accuracy than static activities but lower accuracy than walking. Both sit-down and stand-up are motion activities and they have different impacts on CSI patterns, so they are easier to recognize compared with static activities. The issue for transition activities is that they are sometimes misclassified as walking. Some testing persons have some different static/-motion patterns compared with other persons. For example, one person could be playing with a smartphone during data collection. This introduces minor movements which could confuse the recognition algorithm to misclassify transition activities as walking, as shown in Fig. 4.11. Therefore, the major issue is how to improve the recognition performance of static and transition activities.

The recognition accuracy of static and transition activities is significantly improved by the state machine, as shown in Fig. 4.11. This is achieved by using the state machine

to learn time dependencies and context information from neighboring CSI segments. The recognition algorithm utilizes the relatively high accuracy of transition activities and the temporal dependencies to improve the recognition performance of static and transition activities. Person 5 and 7 have slightly lower recognition accuracy than other persons. For person 5 and 7, 14% to 20% of sitting are misclassified as standing, and 11% to 17% of sit-down are misclassified as standing. The major reason is that the state machine sometimes gives wrong classification results of the temporal dependency information. The state machine of the proposed design is a small 1D CNN with fixed neural architecture. It is possible to improve the recognition performance by using a deeper 1D CNN as the state machine and with its neural architecture optimized by the reinforcement learning agent. We leave the optimization of the state machine as future work.

Table 4.5: Number of parameters and inference time consumption per instance for different convolution types.

Convolution Type	State Machine	Number of Parameters		Inference Time Consumption per Instance	Average Accuracy (std)
		Trainable	Non-Trainable		
Conv2D	None	426324	82	18.5 milliseconds	69.14% (6.8%)
Conv3D		127167	65	20.6 milliseconds	71.29% (4.0%)
SeparableConv2D		7992	72	18.7 milliseconds	76.86% (2.3%)
Conv2D	Conv1D	7500	50	11.3 milliseconds	92.57% (5.5%)
Conv3D		39911	60	14.3 milliseconds	83.29% (9.9%)
SeparableConv2D		13743	46	12.2 milliseconds	96.60% (2.0%)

Table 4.5 shows the details of the trained model with and without the state machine. SeparableConv2D has a higher recognition accuracy and comparable inference time consumption as Conv2D and Conv3D for both with and without the state machine. The inference time consumption per instance is calculated by running the trained CNN on each testing CSI instance one by one. Non-trainable parameters are from batch normalization layers. These parameters are updated with the mean and variance of the batch normalization input, but are not trained with backpropagation.

Impact of Reinforcement Learning

To check the impact of reinforcement learning, we use SeparableConv2D as the recognition algorithm with or without reinforcement learning for neural architecture search. Fig. 4.2 shows the average recognition accuracy of SeparableConv2D with or without reinforcement learning. When the location and orientation of the WiFi receiver is known, the accuracy is 99% if the data of testing persons are seen during training. But the accuracy drops to 84% for leave-one-person-out validation wherein the data of testing persons are not seen during training. When the location and orientation of the WiFi receiver is unknown during training, the recognition accuracy of SeparableConv2D drops to 62%. When reinforcement learning is used for neural architecture search, the recognition accuracy of SeparableConv2D is improved to 77% for unknown receiver locations and orientations. When Conv1D is added as the state machine, the accuracy of SeparableConv2D with reinforcement learning is improved to 97%.

4.4.2 Evaluation Results of New Datasets and New Environments

This section presents evaluation results of two public datasets from S.Yousefi-2017 [205] and FallDeFi [110].

Dataset Overview

A summary of the two datasets is shown in Table 4.3. For S.Yousefi-2017 [205], CSI measurements are collected in 1 room from 6 persons with 1 transmitter and 1 receiver. For FallDeFi [110], there are 6 rooms, 5 persons, 5 transmitters and 5 receivers. There are 2079 and 397 instances for S.Yousefi-2017 [205] and FallDeFi [110], respectively. The size of each instance for both datasets is (2000, 3, 30) representing 2000 CSI matrices with 3 receive antennas and 30 subcarriers measured in 2 seconds. Both datasets have CSI measurements of other activities, like fall, bend and pickup, but only 5 activities, i.e., sitting, standing, sit-down, stand-up and walking, are included in the evaluation. More details of these two datasets can be found in [205] and [110].

These two datasets only have ground truth labels for sitting, standing and walking but do not have ground truth labels for transition activities including stand-up and sit-down.

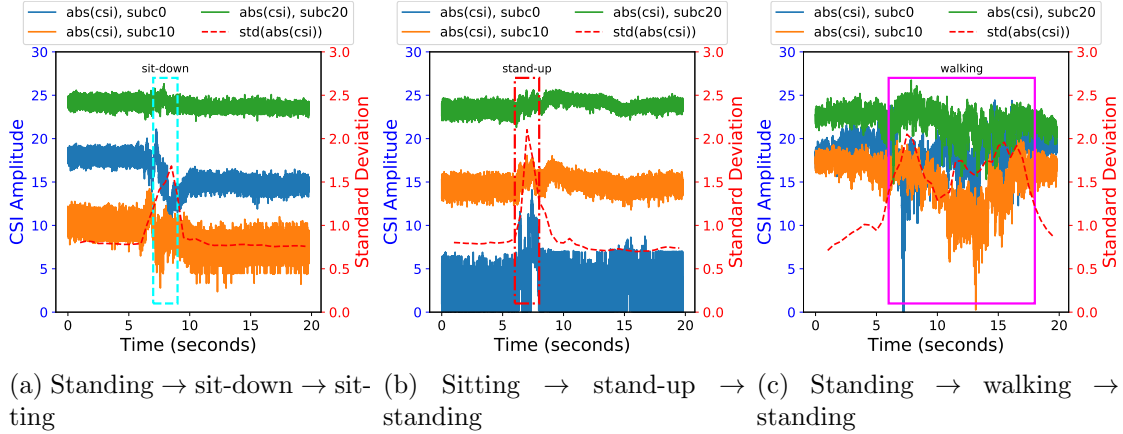


Figure 4.12: CSI segmentation of different activities using standard deviation of CSI amplitude. CSI data is from S.Yousefi-2017 [205].

To address this issue, we use the standard deviation of CSI amplitude to calculate the start and end time of transition activities. This is similar to the method used in FallDeFi [110] for CSI segmentation. Fig. 4.12 shows some examples of CSI segmentation of different activities. The standard deviation is calculated by the amplitude of CSI measurements within a 2 seconds time window with a sliding window of 0.5 second. The start and end of transition activities are calculated by comparing the standard deviation with a pre-defined threshold which is $0.8 \times \max(std(csi))$ in our case. Based on the ground truth label of the CSI sequence, the ground truth labels of the calculated transition window and the CSI segments before and after the transition window can be determined. For example, because the ground truth label of the CSI sequence in Fig. 4.12a is known as sitting, after the transition window is detected within the time window from 5 seconds to 7 seconds, the labels can be determined as standing before 5 seconds, sit-down from 5 seconds to 7 seconds, and sitting after 7 seconds. The labeled CSI segments are fed to different neural networks for evaluation.

Evaluation Results

The evaluation results of new datasets and new environment are shown in Fig. 4.13. The recognition accuracy of the proposed design, i.e., SeparableConv2D with state machine and reinforcement learning, is 80% and 83% for S.Yousefi-2017 [205] and FallDeFi [110],

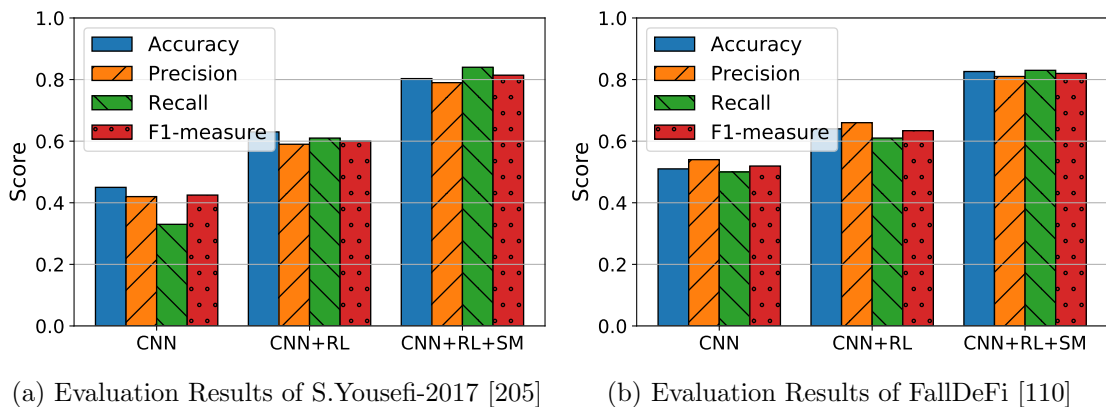


Figure 4.13: Evaluation results of new datasets and dew environments: (a) test of unseen persons, (b) test of unseen environments, unseen persons, and unknown locations and orientations of the WiFi transmitter and receiver.

respectively. For SeparableConv2D with reinforcement learning but without Conv1D as the state machine, the accuracy drops to 63% and 64% for S.Yousefi-2017 [205] and FallDeFi [110], respectively. For SeparableConv2D without state machine or reinforcement learning, the accuracy further drops to 45% and 51%. Although the accuracy of the proposed design for these two datasets is 14% to 17% lower than when it is evaluated by our dataset, the reinforcement learning agent and state machine provide similar accuracy improvements for all the three datasets.

4.5 Discussions

4.5.1 Overhead of Neural Architecture Search

The neural architecture search agent, NASCell, has very high overhead. For computer vision tasks, NASCell takes hundreds of GPU hours to find a good neural architecture. For our dataset with 1455 CSI instances each with size of (200, 3, 3, 30), NASCell takes about three weeks for a 1080 Ti GPU to find the neural architecture for SeparableConv2D without state machine. For the SeparableConv2D with Conv1D as the state machine, the searching time is about 10 days. Recently, there are some new approaches for more efficient

neural architecture search, such as Efficient Neural Architecture Search (ENAS) [115] and Differentiable Architecture Search (DARTS) [84], for computer vision tasks. Compared with NASCell, these two approaches has about 1000 times less overhead with comparable accuracy. These efficient neural architecture search approaches can also be used to find the suitable neural architecture for CSI data. Moreover, since CSI is different from images, new neural architectures and new neural architecture search methods are needed to design and search for the best neural network models that are specifically designed for CSI data.

4.5.2 Robustness in New Environments

The accuracy of the proposed design is 97% for our dataset, and it drops to 80% and 83% for S.Yousefi-2017 [205] and FallDeFi [110], respectively. The major reasons of lower accuracy for the two public datasets are the dataset size and the quality of CSI segmentation and labeling. There are 14555 instances for our dataset, while there are only 2079 and 397 instances, respectively, for S.Yousefi-2017 [205] and FallDeFi [110]. Increasing the dataset size should also bring performance improvements. In our dataset, motion tracking devices and audio instructions are used to calculate fine-grained CSI segmentation and accurate ground truth labeling. But the two public datasets do not have accurate labeling information for transition activities, which could also impact the recognition performance. Besides, the length of CSI sequences of the two datasets is less than 10 seconds, so these two datasets contain less temporal dependency information for the state machine to learn. Collecting more CSI data with fine-grained labeling information and enough temporal dependency information could improve the performance of the proposed design.

4.6 Chapter Summary

In this chapter, we propose a novel deep learning solution for robust activity recognition with WiFi. The proposed solution uses a 2D CNN as the recognition algorithm, a 1D CNN as the state machine, and a reinforcement learning agent to find the best neural ar-

chitecture for the recognition algorithm. We evaluate the proposed design with real-world traces of 14555 instances of 5 activities performed by 7 persons. The proposed design provides 97% average recognition accuracy for unknown receiver locations/orientations and for unseen persons. The reinforcement learning agent provides 15% accuracy improvement compared with when the neural architecture is manually searched. The state machine, along with the reinforcement learning agent, provides another 20% accuracy improvement by learning temporal dependencies from history classification results. The proposed design requires very little human efforts for ground truth labeling, signal processing, feature engineering, parameter tuning, and neural architecture search.

Chapter 5

RoFi: Rotation-Aware Channel Feedback for WiFi

5.1 Introduction

WiFi has a very rapid growth with the increasing popularity of wireless devices and the growing demands of wireless data traffic. Multiple-Input Multiple-Output (MIMO) is one of the key technologies for WiFi to achieve high throughput. Specifically, 802.11n employs single-user MIMO to improve the receiving Signal-to-Noise Ratio (SNR) and data rates [61]. 802.11ac uses multi-user MIMO, which allows transmitting multiple packets concurrently to different receivers, to further improve throughput [62]. Both 802.11n and 802.11ac employ transmit beamforming to improve SNR by concentrating radio energy on the targeted receivers. Furthermore, MIMO provides Channel State Information (CSI) per sub-carrier, which is used for combating multi-path and frequency-selective fading effects, to accurately predict Packet Delivery Ratio (PDR) and select the best transmission strategies [50, 46].

However, CSI introduces high measurement and feedback overhead for WiFi, especially for mobile and handheld devices. The WiFi Access Point (AP) needs CSI measurements and feedback to calculate the beamforming matrix and select the best transmission strate-

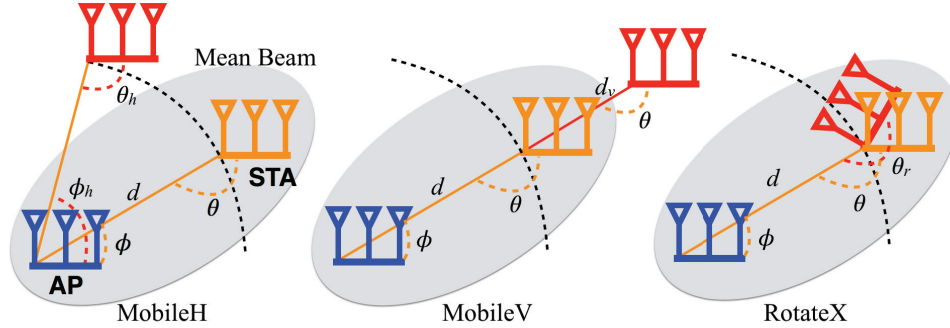


Figure 5.1: Transmit beamforming is impacted differently when the STA is in different mobility scenarios.

gies. The transmission time for data packets is dramatically sacrificed for sending CSI and control packets, since the size of CSI grows rapidly as the number of antennas and channel width increase. Multi-user MIMO has even higher overhead since it needs higher frequency of CSI measurements and feedback to deal with inter-user interference [34]. Moreover, the WiFi station (STA) consumes much energy for sending CSI feedback to the AP. CSI feedback overhead accounts up to 91% of the total energy consumption of WiFi receivers¹. Thus it is crucial to eliminate unnecessary CSI feedback, especially for mobile and handheld devices, because they are typically battery powered.

For WiFi networks with transmit beamforming enabled, the AP needs to steer the signal to the direction of the STA, so it has different feedback requirements if the STA is in different mobility scenarios. For instance, the AP does not need frequent CSI feedback for the STA that is only rotating, such as a mobile device running games that only require device rotation. As shown in Fig. 5.1, the distance and Angle of Departure (AoD) between the AP and STA do not change if the STA is rotating along the X axis (marked as RotateX, shown in Fig. 5.4b), but either one changes if the STA is moving vertically (MobileV) or horizontally (MobileH) to the circle around the AP. The AP has very different CSI feedback requirements when the STA is rotating compared with when it is moving or static.

If the STA sends CSI feedback only when it is needed, the CSI feedback overhead can

¹The result is calculated by energy consumption measurements of the Intel 5300 WiFi chipset with data packet of 1,500 bytes. The calculation and parameter settings are shown in equation (5.13).

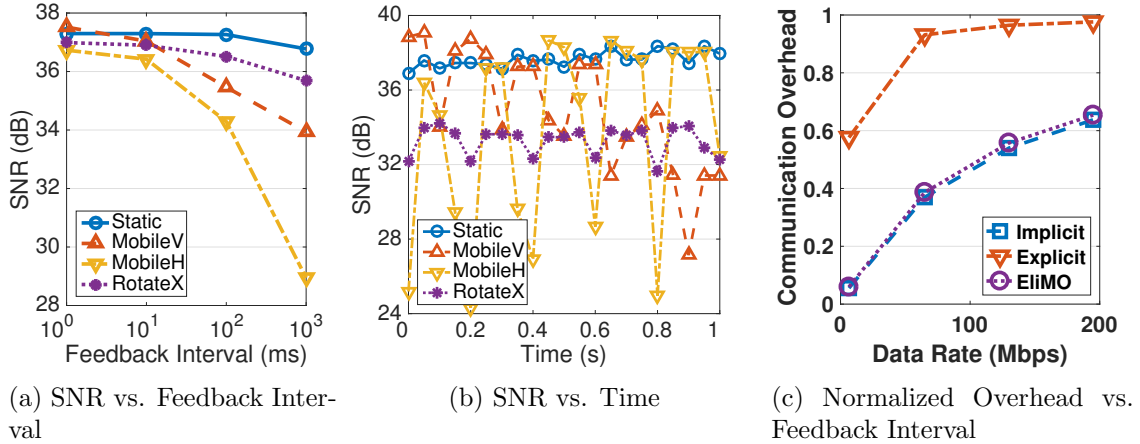


Figure 5.2: SNR results of the STA show different feedback requirements when the STA is in different mobility scenarios.

be significantly reduced while maintaining high throughput. Fig. 5.2a shows SNR results of the STA with different feedback intervals in different mobility scenarios. For RotateX, the AP is able to tolerate long feedback intervals with negligible SNR decrease for the STA. Besides, the STA has more stable SNR variations when it is rotating than when it is moving, as shown in Fig. 5.2b. If the STA is rotating, the normalized overhead, which is computed as the ratio of transmission time for control packets to the total transmission time, can be reduced by 85-94% by using feedback interval of 100ms, as shown in Fig. 5.2c. Therefore, different feedback intervals and transmission strategies should be used if the STA is in different mobility scenarios.

There are many mobile and Internet-of-Things (IoT) systems that require wireless connections and device rotation at the same time. For example, some wireless cameras need to rotate to get a better view angle. Home and industrial robots need rotation for certain tasks. Wireless Virtual Reality (VR) devices sometimes require the user to rotate the headset or handheld controller. Wireless drones sometimes rotate because of in-device or remote control commands; remote controllers/monitors of drones also rotate in some cases. We run four racing and simulation games on an Android smartphone and show the percentage of the running time of different mobility types in Fig. 5.3b. The total

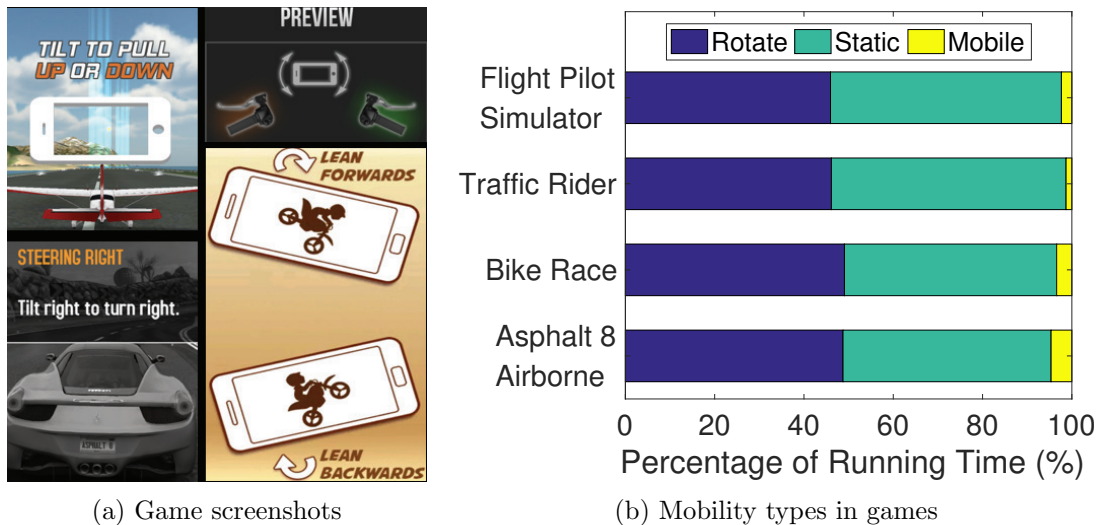


Figure 5.3: Some game applications need users to rotate the device. (a) Games that require device rotation. Top-left: Flight Pilot Simulator; top-right: Traffic Rider; bottom-left: Asphalt 8 Airborne; bottom-right: Bike Race. (b) Percentage of running time of each mobility type for each game.

running time for each game is about 20 minutes. The mobility status of the smartphone is detected by the geomagnetic field sensor and accelerometer every 5 milliseconds. For each game, the device is in the rotation state for about 50% of the running time. Thus, it is necessary to distinguish whether the device is rotating, considering different CSI feedback requirements in different mobility scenarios.

Existing mobility-aware metrics, like CSI similarity [46, 143], Time-of-Flight (ToF) [143, 99, 98, 35], and compression noise [195], cannot distinguish rotation from other mobility scenarios. CSI similarity and ToF are used for mobility-aware rate selection in [143]. Experiments show no significant difference for CSI similarity in rotation and mobile scenarios. ToF results are also similar when the STA remains static, rotates locally, or moves horizontally to the circle around the AP, since the distance between the AP and STA does not change for these three scenarios. Compression noise is used to adjust feedback compression levels for mobile and static scenarios in [195], but experiments show indistinguishable compression noise results for rotation and mobile scenarios. For these three metrics, the AP still needs per-packet CSI feedback if the STA is rotating. Therefore, ro-

tation detection is needed to eliminate unnecessary CSI feedback. The challenge is how to detect STA rotation just based on CSI and how to give efficient CSI feedback in different mobility scenarios.

We propose RoFi, rotation-aware WiFi channel feedback, to eliminate unnecessary CSI feedback by addressing this challenge. RoFi uses Power Delay Profile (PDP) similarity to distinguish device rotation from other mobile scenarios. The STA sends CSI to the AP with the proper feedback interval according to the mobility detection result. The STA calculates the Power of the Strongest Path (PSP) from PDP to refine CSI feedback when the STA is detected in the status of rotation and static. The AP calculates the beamforming matrix and selects the data rate based on the most recent CSI feedback. In summary, we make the following contributions:

- We conduct CSI measurements and show that the AP has different CSI feedback requirements when the STA is in the mobility status of rotation, mobile, or static.
- We show the failure of CSI similarity, ToF, and compression noise, in distinguishing rotation from other mobility scenarios. Therefore, we propose PDP similarity to detect the mobility status of the STA by just using CSI.
- We present rotation-aware CSI feedback to reduce unnecessary CSI feedback with negligible SNR decrease. It improves the performance and efficiency of WiFi STAs.

RoFi does not need frame format modifications and is compatible with legacy 802.11 protocols. RoFi is evaluated with CSI traces collected in different mobility scenarios. Performance metrics, including overhead, throughput, and energy consumption, are used to compare RoFi with state-of-the-art feedback compression and rate selection algorithms. For fixed data rates, RoFi reduces 7-38% feedback overhead in different mobility scenarios, and the maximum SNR decrease introduced by RoFi is lower than 1dB. RoFi also provides up to 52% higher throughput and 48% lower energy consumption. In rotation scenarios, with rate selection enabled, RoFi has up to 22% higher throughput and 47% less energy consumption than existing rate selection algorithms that do not use CSI.

The rest of the chapter is organized as follows. §5.2 summaries related works. §5.3 gives the motivation of RoFi with SNR measurements in different mobility scenarios. §5.4 presents the RoFi design, including rotation detection and rotation-aware CSI feedback. Evaluation results of overhead, throughput, and energy consumption are shown in §5.5. §5.6 summaries the chapter.

5.2 Related Work

5.2.1 CSI Feedback Compression

The 802.11 protocol allows feedback compression by sharing the same CSI for multiple packets or sub-carriers, or representing each CSI value with less bits of data [61, 62, 195]. For example, Intel 5300 only reports CSI for 30 sub-carriers with each entry represented by 16 bits [51], while the default CSI requires 32 bits each for 52 sub-carriers for a 20MHz channel. Different quantization techniques [162] can be used to reduce the size of the CSI matrix. CSI-SF [22] predicts multi-stream CSI values using CSI of single-stream packets to reduce CSI sampling overhead. AFC [195] adaptively selects compression levels based on the SNR decrease caused by compression noise. But it does not distinguish whether the receiver is rotating or moving and requires per-packet feedback for both cases. Thus, it fails to eliminate unnecessary CSI feedback if the STA is rotating. RoFi provides CSI feedback only when it is needed by rotation-aware channel feedback.

5.2.2 MIMO Rate Selection

There are many works on WiFi rate selection, where the data rate is determined by channel width, antenna selection, code rate, and modulation scheme. Each data rate selection has the maximum rate and the corresponding PDR it can delivery. The problem is how to select the rate index satisfying certain requirements, such as high throughput, low delay, low energy consumption, etc. A simple yet effective solution is to predict the PDR based on per-packet SNR and the PDR-SNR curve [128]. For MIMO, the SNR-based

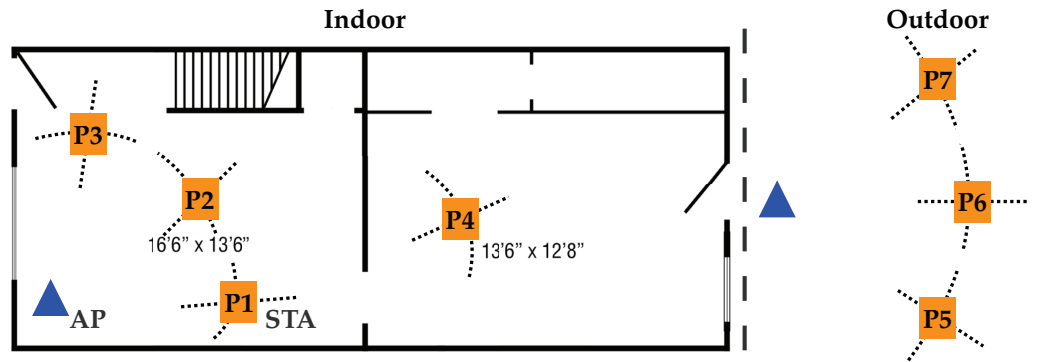
algorithm performs poorly since the PDR-SNR model is not accurate due to frequency-selective fading effects. Effective SNR [46, 50] accurately predicts PDR using CSI, instead of per-packet SNR, and provides high throughput for MIMO networks. But it needs to measure and exchange CSI continuously, introducing huge measurement and feedback overhead. The Linux WiFi driver uses PDR-based rate selection that measures PDR by probing packets every 50ms [101, 59]. The PDR-based algorithm has high probing overhead. It is not suitable for mobile environments since the MIMO channel changes quickly during the 50ms measurement period.

5.2.3 Mobility-Aware WiFi Protocols

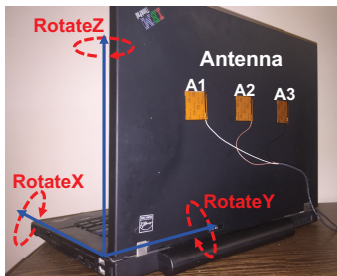
Sensors are used to enhance WiFi protocols by providing movement information [127], but it only provides boolean movement hints and requires modifications of WiFi frame formats and protocols. CSI similarity is used to enable mobility-aware rate selection in [143]. The aforementioned mobility-aware methods are not able to distinguish whether the STA is in the status of rotation or mobile. CSI provides detailed information of attenuation and phase shifts [148, 36] to calculate AoA and Time-of-Flight (ToF) in decimeter-level accuracy [75, 151]. AoA and ToF can be used to detect rotation, but it requires extensive CSI measurements from multiple packets and APs [75] or scanning of all available frequency bands [151, 197]. ToF can also be measured by the time interval between data and ACK packets using off-the-shelf WiFi chipsets [143, 99, 98, 35], but the accuracy cannot be guaranteed at nanosecond level, which makes it hard to distinguish whether the STA is rotating. BeamAdapt [206] brings beamforming to mobile devices, and performance considering device rotation is studied. Unlike RoFi considering the STA as the receiver, BeamAdapt uses the STA as the transmitter, and it does not consider the accuracy and overhead of CSI feedback.

5.3 Motivation

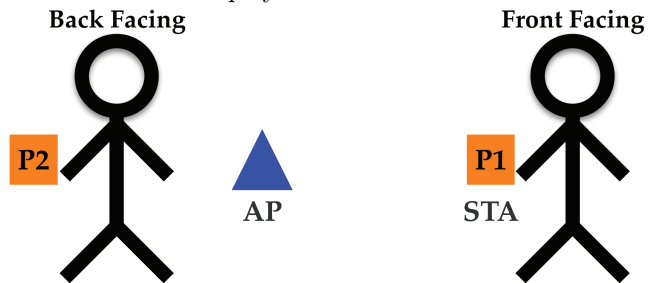
This section presents experiment measurements to analyze receiving SNR of the STA in different mobility scenarios. We found that rotation needs to be separately addressed to eliminate unnecessary CSI feedback.



(a) Indoor and Outdoor Deployments



(b) Rotation Directions



(c) STA Placements w/ or w/o Human Blocking

Figure 5.4: Experiment setup for CSI measurements in different mobility scenarios.

5.3.1 Experiment Setup

We conduct CSI measurements using Intel WiFi Link 5300 and 802.11n CSI tool [51] in various real-world scenarios. Deployment locations of the AP and STA are shown in Fig. 5.4a. Indoor and outdoor experiments are conducted separately, and there is only one AP and one STA at the same time. At each STA position, i.e., P1 to P7, the STA moves vertically (MobileV) or horizontally (MobileH) to the circle around the AP, with the speed of about 1.2 meters per second. The STA rotates along X/Y/Z axis (RotateX/Y/Z), as shown in Fig. 5.4b, or remains static (Static). The rotation speed for RotateX/Y/Z

is about 180 degrees per second. Mobile stands for either MobileV or MobileH, and Rotate represents either RotateX, RotateY, or RotateZ. For each mobility scenario, CSI is measured with or without human blocking, as shown in Fig. 5.4c. CSI measurements for each scenario at each position are repeated for at least 20 times.

The WiFi AP and STA operate at 5GHz, and the channel width is 20MHz. The AP has 3 external antennas. The STA has 3 internal antennas spaced 3 inches apart, which can be installed on smartphones and tablets, as shown in Fig. 5.4b. The transmitting power of the AP is fixed at 17dBm, and there are no other interference sources. The AP continuously sends packets to the STA, which collects CSI measurements about every 0.5 milliseconds. Each received packet has a preamble that contains training symbols for calculating the transmitted signal X . When the STA receives the packet, it gets the corresponding received signal Y . The STA calculates the feedback CSI H_f for each sub-carrier by the MIMO channel model $Y = H_f X + N$, where N is the noise signal. Note that 802.11n CSI tool [51] only provides CSI values of 30 sub-carriers even though a 20MHz WiFi channel has 52 sub-carriers [61, 62, 34, 114]. H_f is sent back to the AP to calculate the beamforming matrix Q for transmitting data packets.

For a data packet, the transmitted signal is QX instead of X . The AP calculates Q as a function of H_f to map X to different spatial streams, so that it can steer the radio signal to the target receivers. In Zero Forcing BeamForming (ZFBF) [49, 195], which is widely used for both single- and multi-user beamforming, the beamforming matrix is $Q = H_f^*(H_f H_f^*)^{-1}$, where $(\cdot)^*$ is the conjugate transpose operation. Now the channel model for data packet transmission is $Y = H_d Q X + N$, where H_d is the CSI matrix measured by the data packet. Note that there is a time interval between H_f and H_d . After receiving Y , the STA uses Minimum Mean Square Error (MMSE) [49, 46, 50] to decode the received signal. The SNR for the k th sub-carrier of the j th spatial stream is $snr_{k,j} = 1/Y_{jj} - 1$, where $Y = (H_k^* H_k + I_S)^{-1}$, $H_k = H_d Q$ is the effective CSI of sub-carriers k for the ZFBF transmitter, and I_S is a $S \times S$ identity matrix with $S = \min(N_t, N_r)$ as the maximum number of streams supported by the MIMO channel [46, 50]. The difference

between H_d and H_f introduces beamforming errors to ZFBF and influences the receiving SNR for the STA. The receiving SNR at time t with feedback interval δ is

$$snr(t, \delta) = db(\sum snr_{k,j}/\sqrt{S}), \quad (5.1)$$

where δ is the time interval between H_d and H_f , and \sqrt{S} is the scaling factor [51].

5.3.2 Measurement Results

We show SNRs in terms of feedback interval and time in different mobility scenarios.

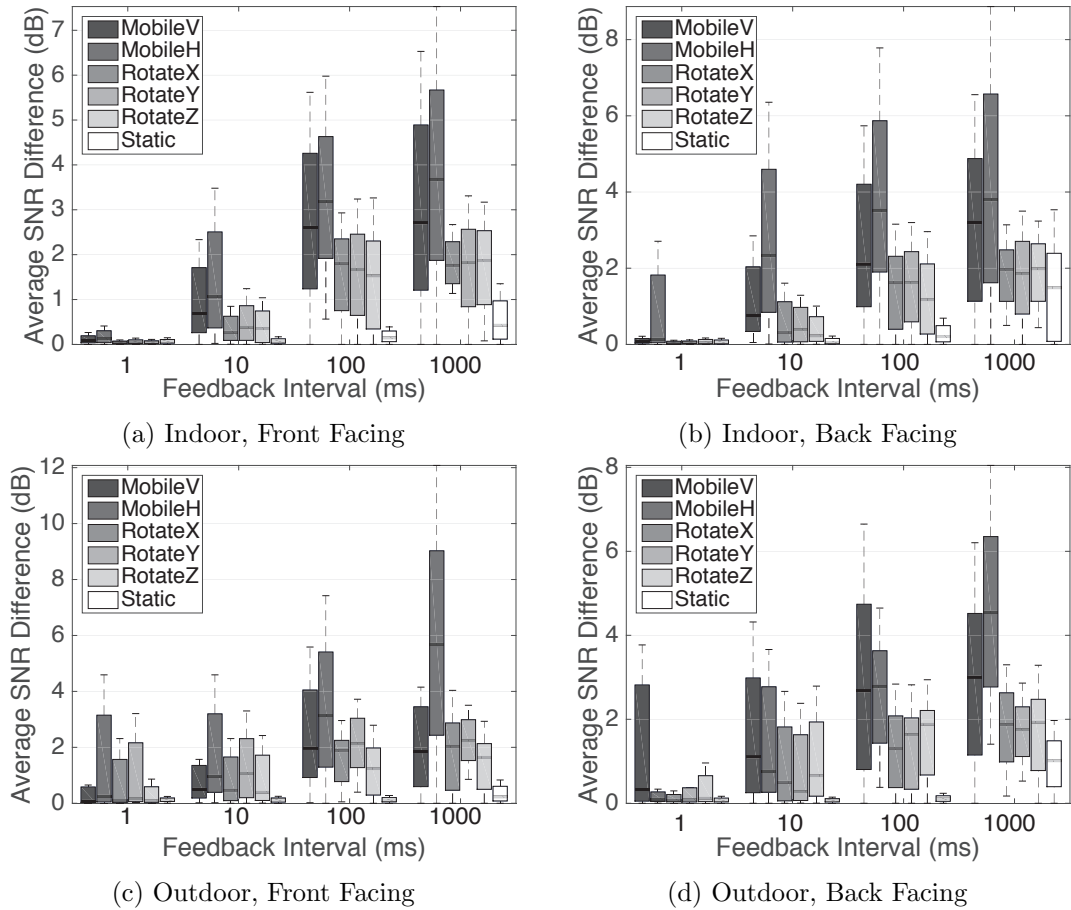


Figure 5.5: Rotate has smaller SNR differences than MobileH and MobileV.

SNR vs. Feedback Interval. Fig. 5.2a shows SNRs with different feedback intervals

in different mobility scenarios. *Rotate has much smaller SNR differences when using long feedback intervals compared with Mobile.* For MobileV, SNR with feedback interval of 1ms is about 3dB higher than that of 10ms. MobileH has 8dB lower SNR when feedback interval is changed from 1ms to 1,000ms. For Rotate, there is no significant SNR difference for feedback intervals less than 100ms. To quantify the impact of feedback intervals, we define SNR difference as

$$snrdiff(t, \delta) = snr(t, 0) - snr(t, \delta), \quad (5.2)$$

where $snr(t, \delta)$ is the SNR at time t with feedback interval δ and is calculated by equation (5.1). Here $snr(t, 0)$ represents the optimal SNR at time t without feedback delay, which means that H_f and H_d are measured at the same time, i.e., $H_f = H_d$.

Fig. 5.5 shows the average SNR difference for different mobility scenarios. The average SNR difference for Rotate is less than 2.1dB. For a certain feedback interval, Rotate has much smaller SNR differences than Mobile. Thus, if the STA is rotating, it should choose a long feedback interval, e.g., 100ms, to reduce CSI feedback overhead. In other words, rotation-awareness could significantly reduce feedback overhead with negligible SNR decrease.

SNR vs. Time. Fig. 5.2b shows SNR variations over time in different mobility scenarios. *Rotate has more stable and predictable SNR variations compared with Mobile.* Both MobileV and MobileH have random and large SNR variations. At 0.6s, for example, the SNR after 50ms changes 7dB and 9dB, respectively, for MobileV and MobileH. However, SNR variations are within 1dB and 2dB for Static and RotateX, respectively. To quantify statistical results of SNR variations, we define SNR variation as

$$snrvari(t, \delta) = |snr(t + \Delta t, \delta) - snr(t, \delta)|, \quad (5.3)$$

where $snr(t, \delta)$ is the SNR at time t with feedback interval δ , and Δt is the time interval

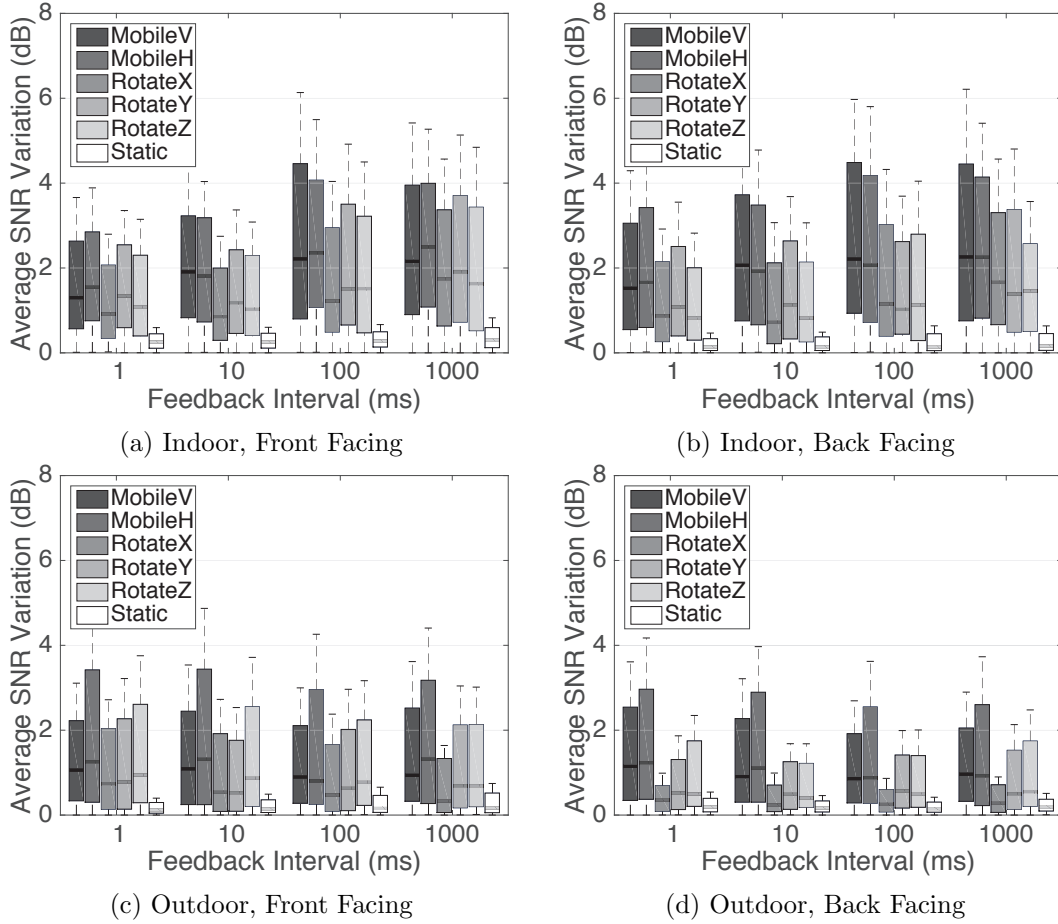


Figure 5.6: Rotate has more stable SNR variations than MobileH and MobileV.

between two SNR measurements.

Statistical results of SNR variations of different mobility scenarios are shown in Fig. 5.6. The measurement interval Δt is 50ms. For Indoor, the average SNR variation of Rotate is about 0.5-1dB lower than that of Mobile. The average SNR variation of RotateX is 1dB lower than that of Mobile for feedback interval of 100ms. For Outdoor, SNR variations of both Rotate and Mobile are smaller than that of Indoor. The average SNR variation for Mobile and Rotate slightly increases for Indoor but remains almost the same for Outdoor, as the feedback interval increases from 1ms to 1,000ms.

To sum up, Rotate has smaller SNR differences and SNR variations than Mobile. WiFi should select different CSI feedback and transmission strategies for Rotate and

other mobility scenarios to improve the performance and efficiency of WiFi STAs. For this purpose, we are motivated to propose RoFi: Rotation-aware WiFi channel feedback.

5.4 RoFi Design

This section presents RoFi design and how it can be used to optimize feedback compression and rate selection.

5.4.1 RoFi Overview

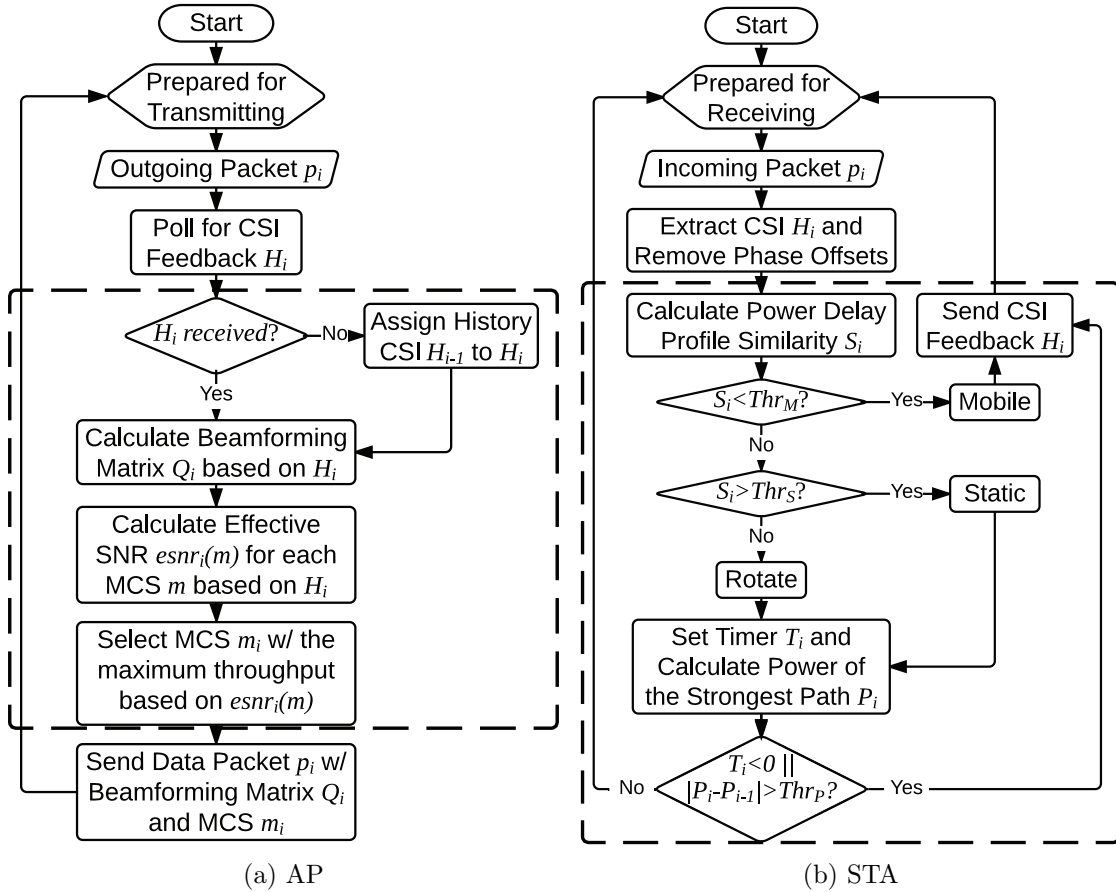


Figure 5.7: RoFi design with added components in dashed rectangles.

The overview of RoFi design for the AP and STA is shown in Fig. 5.7. When the AP has an outgoing data packet p_i for the STA, it first notifies the STA to measure the current CSI

H_i and then polls CSI feedback from the STA. If the AP does not receive the CSI feedback, it assigns history CSI H_{i-1} as the current CSI H_i . The AP calculates beamforming matrix Q_i and effective SNR $esnr_i(m)$ for each Modulation and Coding Scheme (MCS) index m based on H_i . The AP selects the MCS index m_i with the maximum throughput based on $esnr_i(m)$. Finally, the AP sends the data packet to the STA using beamforming matrix Q_i and MCS index m_i .

The STA extracts CSI H_i from the CSI measurement packet. Based on H_i , the STA calculates Power Delay Profile (PDP) similarity S_i to detect whether the STA is in the status of Mobile, Rotate, or Static. If it is Mobile, the STA sends CSI feedback to the AP for each data packet. If it is Rotate or Static, the STA calculates the Power of the Strongest Path (PSP) P_i based on PDP $h_i(t)$. The STA only sends CSI feedback when the change of PSP is larger than a threshold Thr_P , or the time interval since the previous CSI feedback is greater than 50ms and 100ms for Rotate and Static, respectively.

5.4.2 Rotation Detection

Existing Methods. There are three mobility-aware methods using CSI Similarity [46, 11, 143], Compression Noise [195], and Time-of-Flight (ToF) [143, 99, 98, 35]. However, we found that none of these three methods is able to tell whether the STA is in the status of Rotate, as shown in Fig. 5.8. CSI Similarity is calculated as:

$$CS_i = \frac{\sum_{k=1}^{N_s} (h_i(k) - \bar{h}_i)(h_{i-1}(k) - \bar{h}_{i-1})}{\sqrt{\sum_{k=1}^{N_s} (h_i(k) - \bar{h}_i)^2} \sqrt{\sum_{k=1}^{N_s} (h_{i-1}(k) - \bar{h}_{i-1})^2}}, \quad (5.4)$$

where $h_i(k)$ is the CSI magnitude of the k th sub-carrier, and \bar{h}_i is the average CSI magnitude across N_s sub-carriers of the i th packet [46, 11, 143]. CSI Similarity can detect Static, but it can hardly distinguish Rotate from Mobile, as shown in Fig. 5.8a. Compression Noise is defined as:

$$CN_i = \sum_{k=1}^K |(H_i(k) - H_{i-1}(k))(H_i(k) - H_{i-1}(k))^*|, \quad (5.5)$$

where $H_i(k)$ is the CSI value of the k th sub-carrier of the i th packet [195]. Static, Mobile, and Rotate show indistinguishable Compression Noise results, as shown in Fig. 5.8b. The measured ToF tof_m between the data and ACK packet is given by

$$tof_m = 2 * tof_a + t_{SIFS} + t_{ACK}, \quad (5.6)$$

where tof_a is the propagation time of the radio signal, t_{SIFS} is the Short InterFrame Space (SIFS) time between the data and ACK packet, and t_{ACK} is the transmission time for the ACK packet [98, 35]. tof_m is measured by the elapsed time from the departure time of the data packet to the arrival time of the ACK packet. The detail of how to measure tof_m can be found in [98, 35]. Fig. 5.8c shows that the measured ToF is not able to distinguish Rotate from either Static or MobileH.

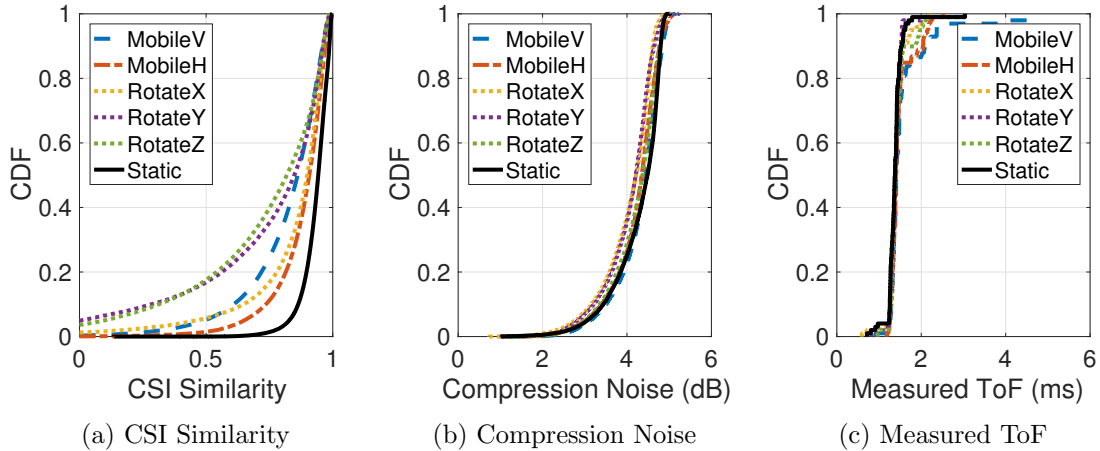


Figure 5.8: Neither CSI Similarity nor Compression Noise is able to distinguish whether the STA is in the status of rotation or mobile.

The Proposed Method. We propose Power Delay Profile (PDP) similarity to detect the mobility status of the STA. Since the AoD and distance (shown in Fig. 5.1) between the AP and STA remain unchanged for Rotate while either one changes for MobileV and MobileH, Rotate and Mobile should have different multi-path fading results. PDP characterizes multi-path channel dynamics of MIMO channels, so PDP similarity provides

better rotation detection results than CSI Similarity, ToF, and Compression Noise.

PDP is the time-domain transformation of channel frequency response by applying Inverse Fast Fourier Transformation (IFFT) on the frequency-domain CSI [197, 134]. The corresponding PDP of CSI $H(f)$ is $h(t) = \sum_{k=1}^K \alpha_k \delta(t - \tau_k)$, where K is the number of paths, α_k and τ_k are the attenuation and delay for path k , respectively. $\delta(\cdot)$ is the delta function. The norm of $h(t)$, $\|h(t)\|_2$, represents the signal strength of each path along which the transmitted signal arrives at the receiver with different time delays. Let $f_i(k) = \|\alpha_k \delta(t - \tau_k)\|_2$ be the signal strength of the k th path of the PDP derived from the i th packet, then the PDP similarity between the i th and $(i - 1)$ th packet is

$$S_i = \frac{\sum_{k=1}^K (f_i(k) - \bar{f}_i)(f_{i-1}(k) - \bar{f}_{i-1})}{\sqrt{\sum_{k=1}^K (f_i(k) - \bar{f}_i)^2} \sqrt{\sum_{k=1}^K (f_{i-1}(k) - \bar{f}_{i-1})^2}}, \quad (5.7)$$

where \bar{f}_i is the average PDP norm of the i th packet.

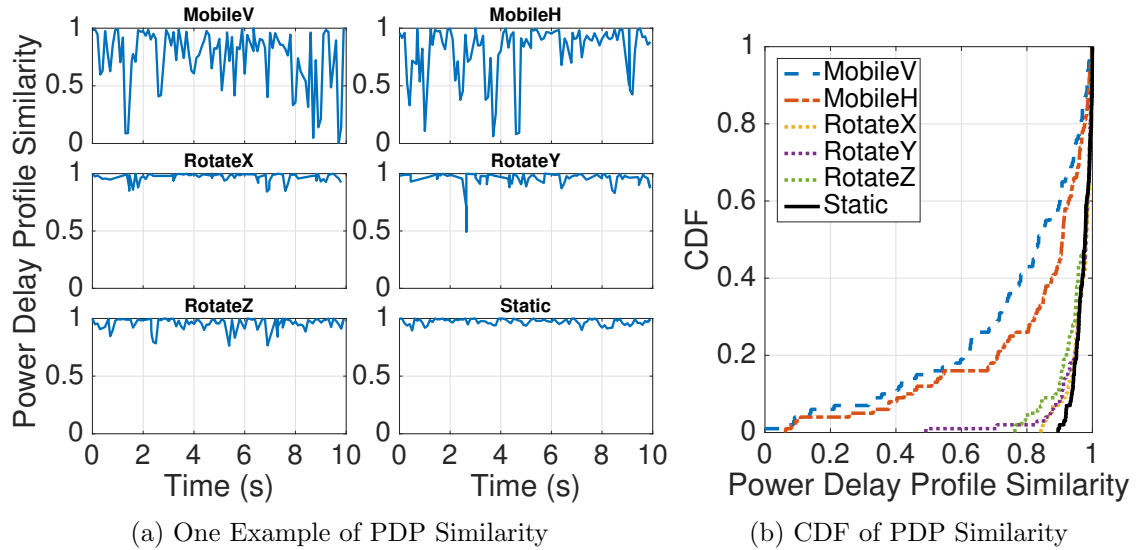


Figure 5.9: Power Delay Profile Similarity of different mobility traces.

Fig. 5.9 shows the CDF of PDP similarity in different mobility scenarios. The time interval between two adjacent packets is 100ms. The PDP similarity for MobileV and MobileH is much lower than that of Rotate and Static. This means that the multi-path

channel of Mobile is less stable than that of Rotate and Static. We use different thresholds of PDP similarity to distinguish Mobile, Rotate, and Static. Since 90% of PDP similarity are larger than 0.9 for Rotate and 0.95 for Static, while 60% are smaller than 0.9 for Mobile, we use the threshold of $Thr_S = 0.95$ and $Thr_M = 0.9$. If the PDP similarity S_i is greater than Thr_S , the STA is detected as Static; if S_i is smaller than Thr_M , the STA is detected as Mobile; otherwise the STA is detected as Rotate.

5.4.3 Rotation-Aware Channel Feedback

The STA determines the CSI feedback interval based on the rotation detection result. For Mobile, the STA sends CSI feedback for each packet. For Rotate and Static, the feedback interval is 50ms and 100ms, respectively. If the rotation detection result is changed, the STA resets the feedback timer T_i to 50ms or 100ms. Otherwise, the STA checks the feedback timer T_i . If $T_i > 0$, the STA changes to receiving state without sending CSI feedback; otherwise the STA sends CSI feedback and transforms to receiving state. The reason for selecting feedback interval of 50ms and 100ms is that it has a good trade-off of feedback overhead and SNR. As shown in Fig. 5.2c, the normalized overhead is significantly reduced using feedback interval of 50ms, but it does not change much when the feedback interval is larger than 50ms. The average SNR decrease for Rotate is less than 2dB by choosing feedback interval 50ms. For Static, the average SNR decrease is less than 1dB for feedback interval of 100ms, as shown in Fig. 5.5.

The AP calculates the beamforming matrix Q_i and selects the MCS index m_i using CSI feedback H_i before sending packet p_i . If no CSI feedback for packet p_i is received, the AP uses history CSI H_{i-1} as the current CSI H_i . In this paper, ZFBF is used as the beamforming algorithm, i.e., $Q_i = H_i^*(H_i H_i^*)^{-1}$. The AP calculates effective SNR (eSNR) [50, 46] for each MCS index using H_i , and selects the MCS m_i with the maximum

achievable throughput by solving

$$\begin{aligned}
& \arg \max_m \quad pdr_i(m) * rate(m), \\
& \text{subject to} \quad pdr_i(m) > Thr_{pdr}(m), \\
& \quad \quad \quad 0 \leq m \leq mMax,
\end{aligned} \tag{5.8}$$

where $pdr_i(m)$ is the Packet Delivery Ratio (PDR) using MCS m calculated before transmitting packet p_i , $Thr_{pdr}(m)$ is the corresponding PDR threshold, $rate(m)$ is the theoretical data rate of MCS m , and $mMax$ is the maximum MCS index. For a 20MHz MIMO channel with three transmitting antennas ($N_t = 3$) and three receiving antennas ($N_r = 3$), the maximum MCS is $mMax = 23$. The AP predicts $pdr_i(m)$ based on the eSNR threshold Thr_{esnr} , above which $pdr_i(m)$ is larger than Thr_{pdr} , i.e., $pdr_i(m) > Thr_{pdr}$ if $esnr_i(m) > Thr_{esnr}$, for each MCS index m . After calculating the beamforming matrix Q_i and selecting the MCS index m_i , the AP sends the data packet to the STA using Q_i and m_i .

The threshold-based rotation detection algorithm sometimes classifies Mobile as Rotate or Static, since PDP similarity of Mobile could be greater than 0.9 in some cases, as shown in Fig. 5.9. Consequently, the STA does not send CSI feedback, while it is needed for the AP. Furthermore, Rotate has small SNR differences and stable SNR variations only during the rotation process but not at the beginning or end of rotation, in which cases CSI feedback is still needed for Rotate. Thus, the STA needs to send CSI feedback to the AP when necessary if the STA is detected as Rotate or Static.

To further refine the aforementioned CSI feedback design, we here define the Power of the Strongest Path (PSP) as $P_i = \max f_i(k)$, $1 \leq k \leq K$, where $f_i(k)$ is the signal strength of the k th multi-path component from the PDP norm of the i th packet, and K is the total number of multi-path components. Fig. 5.10a shows one example of PSP and SNR difference for different mobility traces. For Rotate, *there is a negative relation between PSP and SNR difference*: if PSP remains stable, SNR difference is very low; if

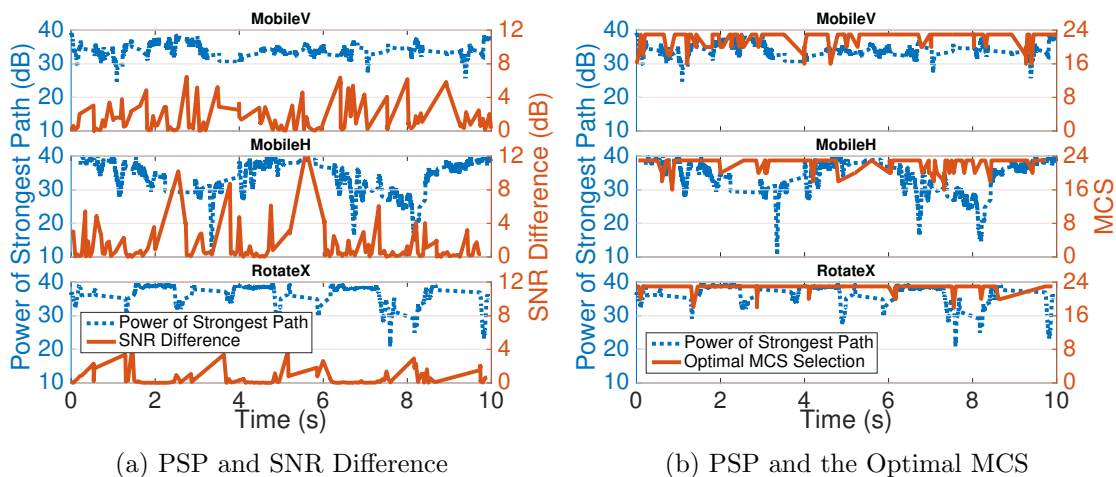


Figure 5.10: PSP is a good indicator of SNR difference and the optimal MCS selection for Rotate traces, but not for Mobile traces.

PSP decreases a lot, SNR difference increases accordingly.

PSP also has a close relation with SNR variation since the strongest path contributes the most to the receiving SNR. Rotate has stable SNR variations, and it should have less frequent rate selection correspondingly. Fig. 5.10b shows the relation between PSP and the optimal MCS selection, which assumes that the AP knows the packet delivery ratio of each MCS at any time and selects the MCS with the maximum throughput. For Rotate, *there is a positive relation between PSP and the optimal MCS selection*: when PSP is at a high level, the optimal MCS selection stays at 23; when PSP drops a lot, it leads to a lower MCS selection.

Based on these two observations, we use PSP to refine CSI feedback when the STA is detected as Rotate or Static. If the PSP change between two adjacent packets is larger than the threshold Thr_P , the STA sends CSI feedback to the AP. PSP is used only if the STA is detected as Rotate or Static, and it does not work for Mobile. Different from Rotate that keeps the STA in the main beam, Mobile changes either the distance or AoD from the AP to STA. For Mobile, there are many variations for SNR difference and the optimal MCS selection even when PSP remains stable, as shown in Fig. 5.10. PSP is not the major factor for Mobile. Therefore, both PDP similarity and PSP are needed so that

CSI feedback is sent only when it is needed.

5.4.4 Overhead Analysis

We present overhead analysis of RoFi to explore potential performance improvements on throughput and energy consumption. Normalized overhead is defined as

$$\tau = t_c / (t_c + t_d), \quad (5.9)$$

where t_c is the transmission time for control packets and t_d for data packets. The AP selects the MCS index m , each with a theoretical data rate $rate(m)$, for each data packet. So $t_d = \sum_{i=1}^N \frac{size(p_i)}{rate(m_i)}$, where N is the number of data packets and $size(p_i)$ is the size of data packet p_i . t_c is calculated as:

$$t_c = \sum_{i=1}^N \left(\frac{size(ctr_i) + size(csi_i)}{rate(0)} + n * SIFS \right) + \sum_{j=1}^M \left(\frac{size(pro_j)}{rate(m_j)} + SIFS \right), \quad (5.10)$$

where $size(ctr_i)$ is the size of control packets, $size(csi_i)$ is the size of CSI, n is the number of SIFSs (Short InterFrame Spacing) for data packet p_i , $size(pro_j)$ is the size of the j th probing packet, and M is the number of probing packets. CSI and control packets, including Null Data Packet Announcement, Null Data Packet, Poll, and ACK, are always transmitted using the lowest data rate, i.e., $rate(m_i)|_{m_i=0} = rate(0)$. The size of CSI is $size(csi_i) = N_t * N_r * N_s * bits(csi) + size(hdr)$, where $bits(csi)$ is the number of bits used for each CSI entry and $size(hdr)$ is the size of packet header. The normalized overhead is significantly reduced when using long feedback intervals, as shown in Fig. 5.2c. RoFi eliminates unnecessary CSI feedback, so the number of CSI packets is much smaller and the normalized overhead is significantly reduced.

The STA spends much less time for CSI and control packets by using RoFi, so it has more time for transmitting data packets to achieve higher throughput, which is calculated

by

$$tpt = \sum_{i=1}^{N'} size(p_i)/(t_c + t_d), \quad (5.11)$$

where N' is the number of received packets. Using long feedback intervals introduces only small SNR decrease if the STA is rotating, as shown in Fig. 5.5 in §5.3. The number of received packets for RoFi is not significantly influenced. RoFi has much smaller t_c , so it provides higher throughput.

RoFi also improves energy efficiency for the STA by sending less CSI packets. Energy efficiency of the STA is evaluated by energy consumption per data bit

$$eb = \frac{\sum_{i=1}^N (er(0) * size(ctr_i) + et(0) * size(csi_i))}{\sum_{i=1}^{N'} size(p_i)} + \frac{\sum_{i=1}^N er(m_i) * size(p_i) + \sum_{j=1}^M er(m_j) * size(proj)}{\sum_{i=1}^{N'} size(p_i)}, \quad (5.12)$$

where $et(m)$ and $er(m)$ stand for energy consumption per bit for transmitting and receiving, respectively, as using MCS index m [48, 131]. For the Intel 5300 WiFi chipset with $et(0) = 90nJ/bit$ and $er(23) = 11nJ/bit$ [48], $size(p_i) = 1,500$ bytes, and $size(csi_i) = 1,872$ bytes, the percentage of energy consumption of CSI feedback is about

$$e_{csi} = 90 * 1872 * 8 / (90 * 1872 * 8 + 11 * 1500 * 8) = 91\%. \quad (5.13)$$

RoFi reduces the number of CSI packets $\sum_{i=1}^N size(csi_i)$ to increase the transmission time for data packets. Besides, $et(m_i)|_{m_i=0}$ for CSI packets is much larger than $er(m_i)$ for data packets [48, 131], so RoFi remarkably improves the energy efficiency of the STA.

5.5 Evaluation

This section shows evaluation results of overhead, throughput, and energy consumption of RoFi compared with existing feedback compression and rate selection algorithms.

Table 5.1: Existing methods for feedback compression and rate selection

Feedback Compression (Fig. 5.11-5.13)	Rate Selection (Fig. 5.14)
CSI Similarity, CoNEXT'14 [143]	SNR-based, SIGCOMM'06 [128]
Compression Noise, MobiCom'13 [195]	PDR-based, Linux Minstrel [101]
Full Feedback, 802.11ac [62]	eSNR-based, SIGCOMM'10 [50]

5.5.1 Evaluation Methodology

The performance of RoFi is evaluated using CSI measurement traces as illustrated in §5.3. Three performance metrics, including overhead, throughput, and energy consumption (equation (5.9), (5.11), and (5.12)), are quantified in different mobility scenarios. Energy consumption parameters, $et(m)$ and $er(m)$ (used in equation (5.12)), for the Intel 5300 WiFi chipset are from [48]. The channel width is 20MHz, and the MCS index m can be selected from 0 to 23 with the data rate ranging from 6.5Mbps to 195Mbps [60]. The size of data packets is 1,500 bytes. The AP uses ZFBF [49, 195] for transmit beamforming and the STA uses the MMSE receiver [49, 46, 50]. The transmitting power is fixed at 17dBm. We compare RoFi with state-of-the-art methods, as shown in Table 5.1, by CSI traces in four mobility scenarios: Mobile, Static, Rotate, and Gaming. The Gaming scenario contains the mobility traces of four games shown in Fig. 5.3. For the Gaming scenario, the ratio of Rotate, Static, and Mobile traces is about 47%, 49%, and 4%, respectively.

Existing feedback compression methods. We compare RoFi with three feedback compression methods: CSI Similarity [46, 143, 11], Compression Noise [195], and Full Feedback [62]. ToF measured by WiFi chipsets has very low accuracy and it provides much worse rotation detection results than CSI Similarity and Compression Noise, so we omit the evaluation of ToF due to space constraints.

CSI Similarity, which is calculated by equation (5.4), is used to detect the mobility status of the STA. The STA sends CSI feedback for each packet if it is moving; otherwise it sends CSI feedback every 100ms. Compression Noise, which is calculated by equation (5.5), is used to calculate the SNR decrease caused by feedback compression. The AP polls for

CSI feedback only if the SNR decrease is large enough to reduce the current data rate. Note that Compression Noise is defined in three domains: time, frequency, and quantization, in [195]. We only use Compression Noise in the time domain since the 802.11n CSI tool [51] provides non-compressed CSI neither in frequency nor quantization domain. The number of sub-carriers is $N_s = 30$ and the number of bits for each CSI entry is $bits(csi) = 16$. There is also a Full Feedback scheme that requires the STA to send CSI feedback for each data packet.

Existing rate selection methods. We compare RoFi with rate selection algorithms based on PDR [101], SNR [128], and eSNR [50, 46]. These rate selection algorithms select the MCS by solving the same problem in equation (5.8), but measure or predict $pdr_i(m)$ differently. The PDR-based algorithm measures $pdr_i(m)$ by probing packets. For probing packets using MCS index m , PDR is calculated by $pdr_i(m) = \alpha * pdr_{t-1}(m) + (1 - \alpha) * pdr_t(m)$, where $pdr_t(m)$ is the PDR measured during the most recent time window and $pdr_{t-1}(m)$ for the previous time window, and α is the averaging weight. It is the default rate selection algorithm for Linux WiFi driver, wherein the time window length is 50ms and the averaging weight α is 0.125 [101, 59].

The SNR-based algorithm predicts $pdr_i(m)$ based on the SNR threshold $Thr_{snr}(m)$ for each MCS index m , i.e., $pdr_i(m) > Thr_{pdr}(m)$ if $snr_i(m) > Thr_{snr}(m)$ [128]. The eSNR-based algorithm uses effective SNR to predict $pdr_i(m)$, which is the same as RoFi, for each packet p_i [50, 46]. Unlike RoFi, the eSNR-based algorithm requires CSI feedback before transmitting each data packet p_i . To avoid unnecessary CSI feedback, the eSNR-based rate selection uses CSI Similarity to detect the mobility status of the STA. If the CSI Similarity is greater than 0.9, the STA sends CSI feedback for each packet; otherwise it sends CSI feedback every 100ms. Both PDR- and eSNR-based rate selections require sending probing packets. There is also an optimal rate selection algorithm. It assumes that the AP knows CSI and PDR for each MCS index at any time and selects the MCS with the highest throughput. Results of the PDR-based algorithm are from real-world measurements, and other rate selection algorithms are calculated from CSI traces.

5.5.2 Performance Results of Feedback Compression

We first compare RoFi with existing feedback compression schemes. Results show that RoFi has lower overhead and energy consumption and higher throughput in different mobility scenarios.

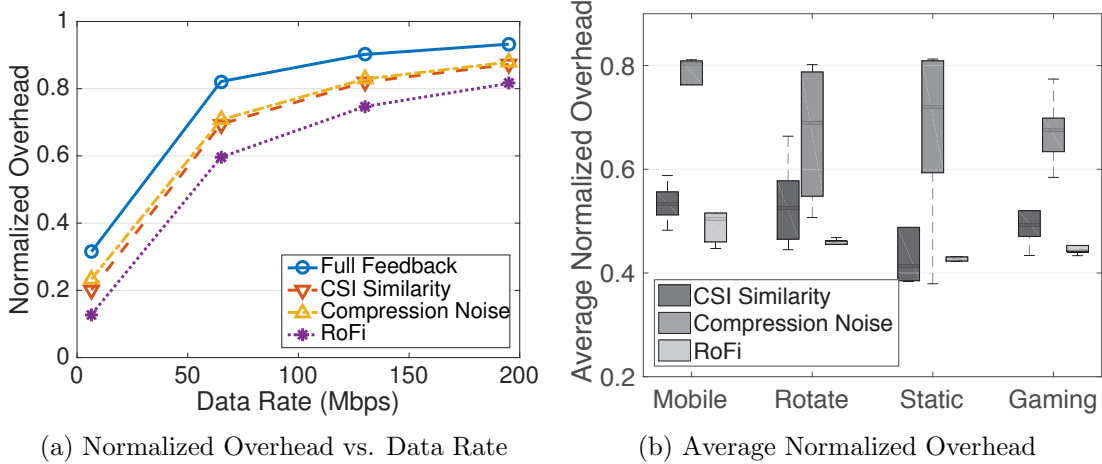


Figure 5.11: Normalized overhead. (a) Using fixed data rates for Rotate. (b) Statistical results for data rate of 65Mbps in different mobility scenarios. The average normalized overhead for Full Feedback is fixed at 0.82 for 65Mbps.

Overhead. Fig. 5.11a shows the normalized overhead, as defined in equation (5.9), using fixed data rates. It is evaluated from the RotateX trace measured at P6 (shown in Fig. 5.4a). Both CSI Similarity and Compression Noise have much higher overhead than RoFi. At data rate of 6.5Mbps, the normalized overhead of RoFi is 0.12, which is only 60% of that of CSI Similarity and Compression Noise. At higher data rates, the normalized overhead of RoFi is 75% of that of CSI Similarity and Compression Noise. In other words, RoFi reduces the transmission time CSI packets by 25-40%. At the same time, there is no obvious SNR difference between RoFi, CSI Similarity, Compression Noise, and Full Feedback. The maximum SNR decrease of RoFi is lower than 1dB.

Statistical results of the average normalized overhead for each mobility scenario are shown in Fig. 5.11b. For Rotate, the normalized overhead of RoFi is 89% and 63% of that of CSI Similarity and Compression Noise, respectively. RoFi also reduces overhead

when the STA is not rotating. The normalized overhead of RoFi is 63% and 60% of that of Compression Noise for Mobile and Static, respectively. For Gaming traces, the normalized overhead of RoFi is 93% and 62% of that of CSI Similarity and Compression Noise. RoFi and CSI Similarity have comparable overhead for Mobile, Static, and Gaming scenarios. The average normalized overhead of Full Feedback is 0.82 for data rate 65Mbps for all mobility scenarios.

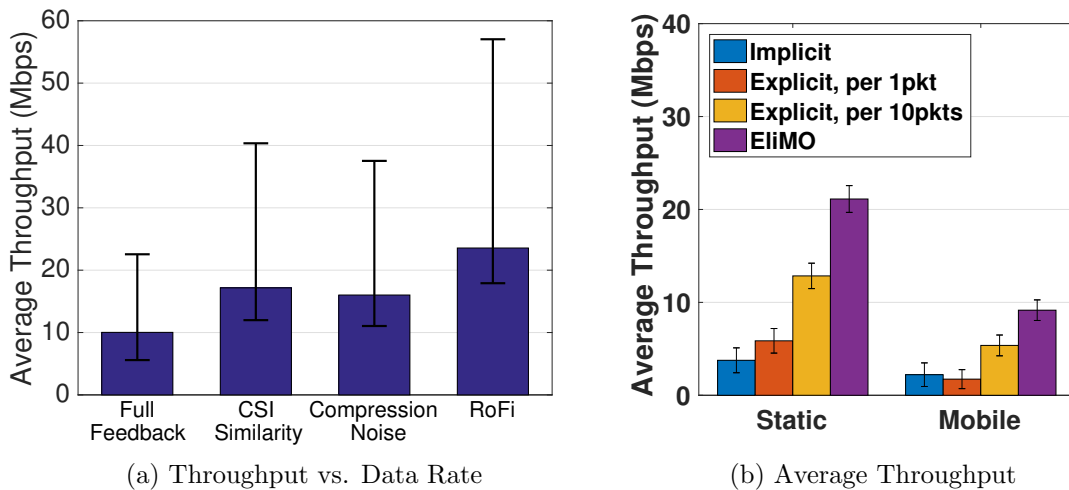


Figure 5.12: Average throughput. (a) Using fixed data rates for Rotate. (b) Statistical results for data rate of 65Mbps in different mobility scenarios.

Throughput. Fig. 5.12a shows throughput, as defined in equation (5.11), for the RotateX trace using fixed data rates. RoFi eliminates unnecessary CSI feedback with negligible SNR decrease, so it provides higher throughput. Full Feedback has the lowest throughput because of sending CSI feedback for each data packet. The throughput of CSI Similarity, Compression Noise, and RoFi is 70%, 60%, and 140%, respectively, higher than that of Full Feedback. Fig. 5.12b shows statistical throughput results for all traces. For Rotate, the throughput of RoFi is $1.52\times$ and $2.16\times$ of that of CSI Similarity and Compression Noise. RoFi has 21%, 43%, and 35% higher throughput than CSI Similarity for Mobile, Static, and Gaming, respectively. RoFi introduces smaller SNR decrease as CSI Similarity, so it still provides higher throughput, even though RoFi has higher normalized

overhead for Static traces as shown in Fig. 5.11b.

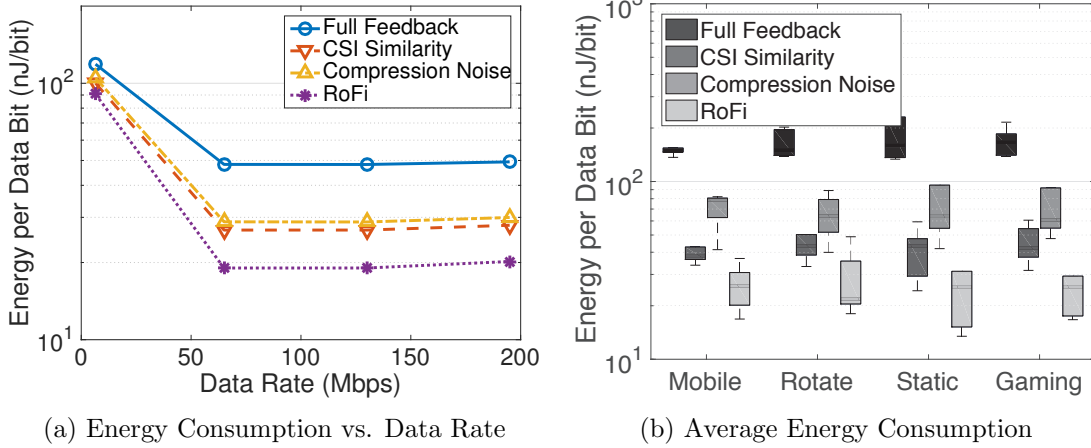


Figure 5.13: Energy consumption. (a) Using fixed data rates for Rotate. (b) Statistical results for data rate of 65Mbps in different mobility scenarios.

Energy Consumption. Fig. 5.13a shows energy consumption, as defined in equation (5.12), for the RotateX trace with fixed data rates. At data rate of 6.5Mbps, energy consumption is almost the same for all feedback compression methods. For data rates of greater than 50Mbps, energy consumption is about $20nJ/bit$ for RoFi, $30nJ/bit$ for CSI Similarity and Compression Noise, and $49nJ/bit$ for Full Feedback. Fig. 5.13b shows statistical results of energy consumption for different mobility scenarios. For Rotate, the energy consumption of RoFi is 48% and 66% lower than that of CSI Similarity and Compression Noise. RoFi consumes less energy by sending less CSI packets for the STA. For Mobile, energy consumption of RoFi is $24nJ/bit$, which is 45% and 53% lower than that of CSI Similarity and Compression Noise, respectively. For Static, RoFi consumes 29% and 69% less energy than CSI Similarity and Compression Noise, respectively. The energy consumption results of Gaming are similar to that of Static.

5.5.3 Performance Results of Rate Selection

Next, we show performance results of RoFi and existing rate selection algorithms based on SNR, PDR, and eSNR. Results show that RoFi has higher throughput and lower energy

consumption in Rotate and Static scenarios.

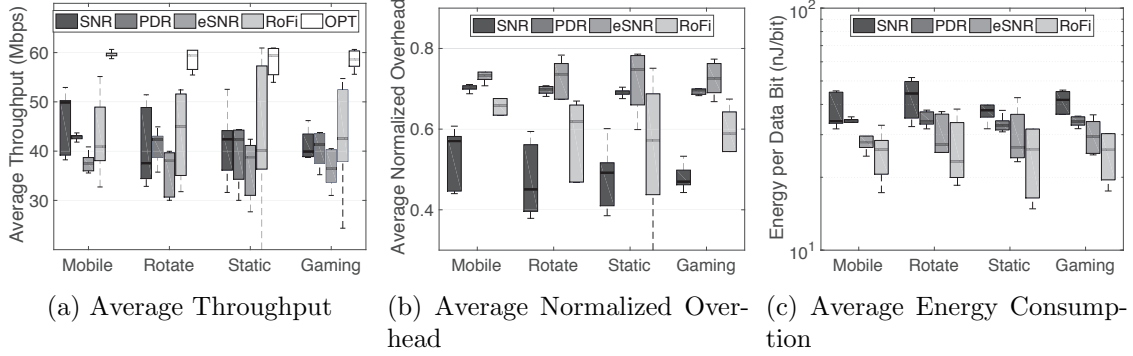


Figure 5.14: Performance results of different rate selection algorithms in different mobility scenarios.

Throughput. Fig. 5.14a shows statistical results of throughput for different mobility scenarios. The throughput of eSNR-based rate selection is the lowest in all mobility scenarios, since it needs extensive CSI measurements and feedback. For Mobile and Static, RoFi has lower throughput than the SNR-based algorithm, since RoFi has much higher normalized overhead as shown in Fig. 5.14b. For Rotate and Gaming, RoFi has 8% and 22% higher throughput than PDR- and SNR-based algorithms, respectively. The reason is that RoFi is able to select much higher data rates with high PDR to send more data packets during the same transmission time. For Static, the average throughput of RoFi is slightly lower than SNR- and PDR-based algorithms. For Gaming traces, RoFi has slightly higher throughput than SNR- and PDR-based rate selections.

Overhead. The results of normalized overhead are shown in Fig. 5.14b. SNR-based rate selection has the lowest normalized overhead in all mobility scenarios, since it does not need CSI feedback or probing packets. The PDR-based algorithm has higher overhead than SNR-based rate selection due to probing packets. The eSNR-based algorithm has the highest overhead since it requires extensive CSI measurements and feedback. The normalized overhead of RoFi is greater than that of SNR-based rate selection, but it is much lower than that of eSNR-based rate selection, in all mobility scenarios. The normalized overhead of PDR- and eSNR-based rate selections is stable across different

Table 5.2: Average energy impact of CSI feedback compression methods

	Energy Impact
Full Feedback	106.45 (100.00%)
CSI Similarity	114.29 (107.37%)
Compression Noise	113.79 (106.90%)
RoFi	112.69 (105.86%)

mobility traces.

Energy Consumption. Fig. 5.14c shows the results of energy consumption in different mobility scenarios. For Mobile and Static, the energy consumption of eSNR-based rate selection is similar to that of RoFi. For Static, the energy consumption of RoFi is 25% and 37% lower than that of SNR- and PDR-based algorithms, respectively. For Rotate, the energy consumption of RoFi is 47%, 31%, and 15% lower than that of SNR-, PDR-, and eSNR-based algorithms, respectively. For Gaming traces, RoFi consumes 43%, 25%, and 17% less energy than SNR-, PDR-, and eSNR-based algorithms, respectively.

5.5.4 Energy Impact of PDP Similarity Calculation

RoFi needs to calculate PDP similarity which may introduce computation overhead for MIMO receivers. In this section, we investigate the energy impact of PDR similarity calculation. We run different CSI feedback schemes, including full feedback, CSI similarity, compression noise, and RoFi, using CSI traces collected in different scenarios. At the same time, we measure the Energy Impact of the simulation process by the Linux command `top`. Energy Impact measures per-process power consumption by CPU usage and wakeup frequency, and it has no physical unit [181]. Fig. 5.15 shows energy impact of four CSI feedback schemes in running time. RoFi has slightly higher energy impact than full feedback, which does not need calculations to determine when to send CSI feedback. The average energy impact as running all CSI traces is summarized in Table 5.2. Compared with full feedback, RoFi only introduces 5.86% extra energy impact. Besides, RoFi has slightly less energy impact than CSI similarity and compression noise.

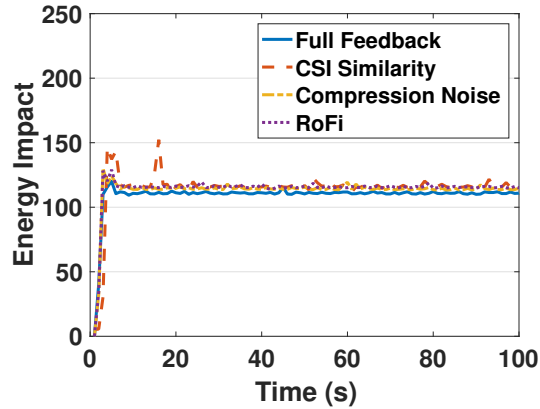


Figure 5.15: Energy impact of different CSI feedback schemes.

5.6 Chapter Summary

This chapter presents rotation-aware WiFi channel feedback to eliminate unnecessary CSI feedback while maintaining high SNR in different mobility scenarios. It firsts shows that WiFi has different CSI feedback requirements when the STA is in different mobility scenarios, including rotation, mobile, and static. Then it shows the failure of existing mobility-aware methods, including CSI similarity, Time-of-Flight, and compression noise, in distinguishing rotation from other mobility scenarios. Finally, it presents RoFi using power delay profile similarity to detect the mobility status of the STA by just using CSI. The STA provides CSI feedback only when it is needed based on rotation detection results. At the same time, RoFi uses the power of the strongest path, which is calculated from power delay profile, to refine CSI feedback when the STA is detected in the status of rotation or static. RoFi brings rotation-awareness to WiFi and helps the AP select the best data rate accurately without extensive CSI measurements and feedback. RoFi significantly improves the performance and efficiency of WiFi STAs in different mobility scenarios by reducing unnecessary CSI feedback with negligible SNR decrease.

Chapter 6

EliMO: Eliminating Channel State Information Feedback from MIMO

6.1 Introduction

Multiple-Input Multiple-Output (MIMO) is the key technology for WiFi to achieve high throughput. Along with Orthogonal Frequency-Division Multiplexing (OFDM), MIMO provides Channel State Information (CSI) per sub-carrier, so that beamforming can be used to improve Signal-to-Noise Ratio (SNR) and throughput [34, 114]. MIMO beamforming provides high throughput by steering the radio energy to the direction of the target receiver, or sending multiple packets concurrently to different receivers. This is done by precoding the transmit signal to different spatial streams and antennas. Moreover, MIMO is able to select the best transmission strategies efficiently assisted by CSI, which helps combat multi-path and frequency-selective fading effects [50, 46].

However, CSI introduces extremely high overhead for MIMO receivers, especially for smart devices like smartphones, smartwatches, and wireless drones. To calculate the beamforming matrix and select the best transmission strategies, the transmitter needs CSI feedback that introduces a lot of computation and communication costs for MIMO receivers. First, MIMO receivers require computation resources to measure and estimate

the CSI matrix. Second, the transmission time for data packets is dramatically sacrificed due to CSI feedback. For channel width of 80MHz, the size of CSI matrix is 1,700 bytes for 3×3 MIMO and 53,000 bytes for 8×8 MIMO [34, 61]. Moreover, multi-user MIMO has even higher overhead since it needs higher frequency of CSI measurements and feedback to deal with inter-user interference [34, 62]. Finally, MIMO receivers consume much energy for sending CSI feedback, which consumes up to 4 times energy as sending a data packet for a MIMO receiver. Thus it is crucial to reduce CSI feedback overhead, especially for smart devices like smartphones, smartwatches, and wireless drones.

There are many methods on reducing CSI feedback overhead [57, 195, 162, 126], but they are not optimized for smart devices and still introduce high computation and communication costs for MIMO receivers. First, MIMO receivers still need to continually measure and estimate the CSI matrix. Second, the STA needs to calculate when to send the CSI matrix and how much feedback is needed, which involves expensive matrix calculations. Finally, MIMO receivers still need to send CSI matrices to the transmitter, even though the feedback frequency and/or feedback size could be reduced. All these computation and communication costs of CSI feedback consume a lot of energy of the STA. The AP can use implicit CSI feedback, which uses the transpose of uplink CSI as the downlink CSI, to reduce feedback overhead [61]. But it has very low beamforming gains, since real-world MIMO channels are not reciprocal due to baseband-to-baseband channels and interferences [61, 114].

We propose EliMO to completely Eliminate CSI feedback from MIMO without sacrificing beamforming gains. EliMO completely eliminates the communication costs of sending the CSI matrix for MIMO receivers. In addition, the computation costs for MIMO receivers are significantly reduced. The challenge is how the WiFi AP could accurately estimate the downlink CSI without explicit CSI feedback. EliMO works as follows. The AP sends Long Training Field (LTF) in the packet header to the STA. The STA inserts the received signal of downlink LTF as Feedback Training Field (FTF) into the packet header and sends it back to the AP. The AP is able to estimate the downlink CSI based

on the received signal of FTF, combined with the uplink CSI that is estimated by uplink LTF sent from the STA. In summary, we make the following contributions.

- We present two-way channel estimation allowing the AP to accurately estimate downlink CSI without explicit CSI feedback.
- We propose Feedback Training Field to completely eliminate CSI feedback without sacrificing beamforming gains.

We evaluate EliMO by experiment measurements in both static and mobile scenarios. Evaluation results show that EliMO is able to provide as low overhead as implicit CSI feedback and comparable SNR as explicit CSI feedback. The average throughput of EliMO is $5\times$ and $4\times$ of that of implicit and explicit CSI feedback, respectively, in static scenarios. In mobile scenarios, EliMO provides $3.6\times/4.5\times$ throughput as implicit/explicit CSI feedback. Energy consumption of EliMO is 85%/91% of that of implicit CSI feedback in static/mobile scenarios. The average energy consumption of EliMO is only 30% and 17% of that of explicit CSI feedback, in static and mobile scenarios, respectively.

The rest of the chapter is organized as follows. §6.2 summaries related works. §6.3 shows the key idea of EliMO and compares it with existing CSI feedback schemes. §6.4 presents the EliMO protocol design, including frame format, two-way channel estimation, and MAC-layer operations. Experiment setup and evaluation results of EliMO for both static and mobile scenarios are given in §6.5. §6.6 summaries the chapter.

6.2 Related Work

There are many papers on reducing the overhead of CSI feedback. IEEE 802.11n/ac protocols allow feedback compression to share the same CSI for multiple data packets or multiple sub-carriers [61, 62]. CSI-SF [22] uses CSI values of one antenna to estimate CSI values of other antennas, which reduces overhead of CSI measurements and feedback. The STA can also use less bits of data for each CSI value [195, 162] to reduce the size of

CSI matrix. AFC [195] adaptively selects feedback compression levels to reduce feedback overhead in different scenarios. Some papers use CSI similarity to detect whether the STA is moving or not, and adjust the frequency of CSI measurements accordingly [143, 11]. This helps to reduce feedback overhead if the STA is not moving. However, all these feedback compression schemes still need CSI feedback from the STA. It introduces high computation and communication overhead for the STA to calculate and send the CSI matrix. Besides, the STA needs to calculate when to send the CSI matrix and how much feedback is needed. This introduces computation overhead for the STA. The calculation and transmission of the CSI matrix consumes a lot of energy for the STA. EliMO completely eliminates CSI feedback and significantly improves the energy efficiency for the STA.

IEEE 802.11n allows implicit CSI feedback [61] to reduce CSI feedback overhead. This is based on the assumption that downlink and uplink channels of the same carrier frequency are reciprocal. But this assumption does not hold in real-world MIMO systems wherein digital baseband channels [61, 114] and interferences are not reciprocal [114]. R2-F2 eliminates CSI feedback for cellular networks [152]. It estimates downlink CSI using uplink CSI at different carrier frequencies. But it does not consider the impact of digital baseband channels, which reduces CSI estimation accuracy seriously. Signpost [224] eliminates CSI feedback for uplink multi-user MIMO. It allows each user to predict its orthogonality to other users by its own CSI. But it only works in the uplink and eliminates CSI feedback from the AP for multi-user MIMO communications. EliMO eliminates CSI feedback in the downlink and reduces computation and communication overhead for the STA. Similar to EliMO, Echo-MIMO [182] also employs two-way channel estimation to eliminate downlink CSI feedback. But it is designed for narrow-band MIMO channels without frequency-selective effects, while WiFi has wide-band MIMO-OFDM channels with frequency-selective effects. It does not consider the impact of digital baseband channels. Besides, Echo-MIMO only focuses on theoretical analysis but does not test with real-world MIMO devices. EliMO is tested with real-world WiFi devices with the impact of wide-band channels, frequency-selective effects, and baseband-to-baseband channels.

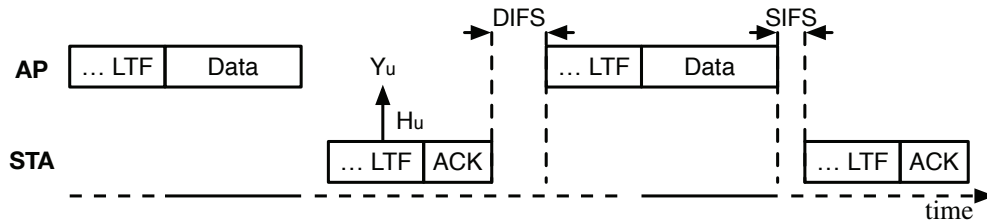
6.3 Motivation

This section presents the background and limitations of existing CSI feedback schemes. The key idea of EliMO is introduced to eliminate CSI feedback without sacrificing beamforming gains. The effectiveness of EliMO is presented by SNR and overhead analysis.

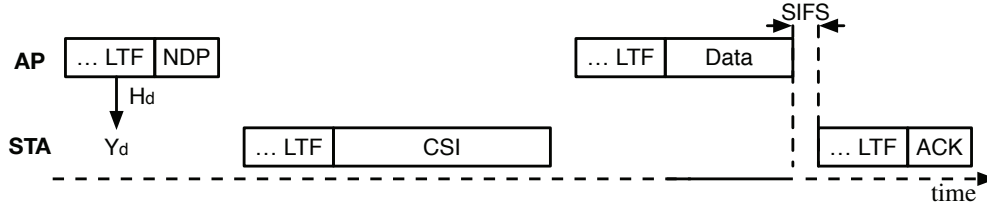
The received signal of beamforming at each sub-carrier is

$$Y = HQX + N, \quad (6.1)$$

where H is the CSI matrix, Q is the beamforming matrix, X is the transmit signal, and N is the noise signal. The beamforming matrix Q , which is usually a function of H , is used for mapping spatial streams to antennas. For Zero-Force BeamForming (ZFBF), which is a widely used beamforming algorithm, the beamforming matrix is $Q = H^*(HH^*)^{-1}$, where $(\cdot)^*$ is the conjugate transpose operation. Due to power constraint of each transmit antenna, the beamforming signal must satisfy $E[|[QX]_j|^2] \leq P_j$, where P_j is the power constraint for the j th transmit antenna [69]. Since the transmitter does not know H , it needs CSI feedback from the receiver.



(a) Implicit CSI Feedback. The AP uses the transpose of H_u as H_d .



(b) Explicit CSI Feedback. The STA sends H_d in the CSI packet to the AP.

Figure 6.1: MAC-layer operations for implicit and explicit CSI feedback.

IEEE 802.11n defines two CSI feedback methods, i.e., implicit and explicit [61], as shown in Fig. 6.1. The CSI matrix is estimated using Long Training Field (LTF), which is part of the packet header. The transmitter sends LTF, which contains predefined signal X , to the receiver. The receiver estimates downlink/uplink CSI H_d/H_u using received signal Y_d/Y_u . For implicit CSI feedback, the AP uses the transpose of H_u as H_d . This is based on the assumption that H_d and H_u are reciprocal. As shown in Fig. 6.1a, the AP measures H_u by the previous ACK packet, and H_u is used to calculate the beamforming matrix for the following data packet. Fig. 6.1b shows MAC-layer operations for explicit CSI feedback. The AP first sends Null Data Packet (NDP) to the STA to measure the downlink CSI. The STA estimates H_d and sends it in the CSI packet back to the AP. The AP calculates the beamforming matrix based on H_d for transmitting the data packet.

MIMO provides high throughput for WiFi networks, but it also leads to high overhead due to Channel State Information (CSI) feedback.

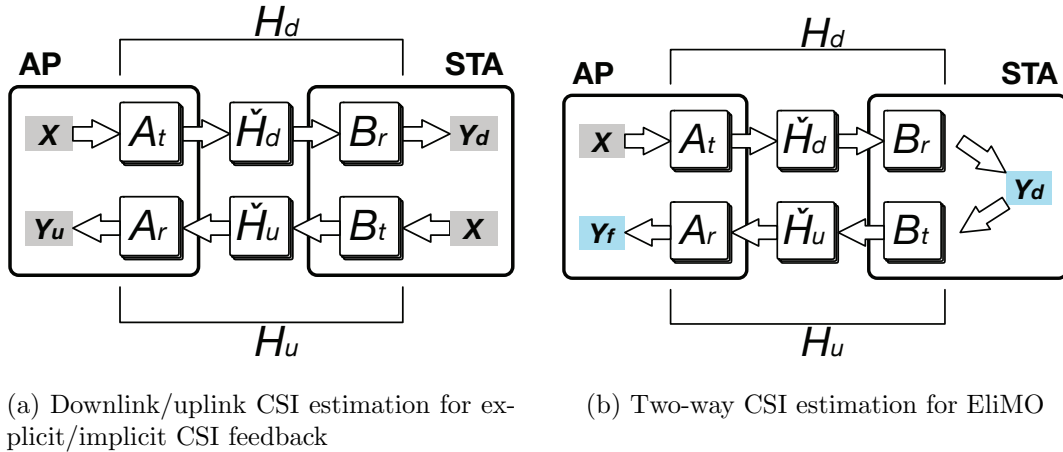


Figure 6.2: Downlink/uplink CSI estimation separately and two-way CSI estimation.

Both implicit and explicit CSI feedback have limitations that influence the performance and efficiency of WiFi STAs. The reason is that they estimate downlink and uplink CSI separately, leading to either low beamforming gains or high overhead. For implicit CSI feedback, the transpose of H_u is not an accurate estimation of H_d , since H_d and H_u are not reciprocal in real-world MIMO systems. As shown in Fig. 6.2a, baseband-to-

baseband channels H_d and H_u are not reciprocal, even though over-the-air channels \check{H}_d and \check{H}_u are reciprocal [61]. Besides, downlink and uplink interferences are usually not reciprocal [114]. The channel reciprocity of multi-user beamforming is even worse due to inter-user interference. For explicit CSI feedback, the STA has very high communication and energy overhead. Since the STA needs to send the CSI matrix to the AP, the STA spends much time and energy for transmitting none-data frames. The communication costs of CSI feedback is very high, since the size of CSI is very large and it grows rapidly as the number of antennas and channel width increase [34, 195].

6.3.1 Key Idea of EliMO

We EliMO to completely Eliminate CSI feedback from MIMO without sacrificing beamforming gains. *The goal of EliMO is to provide as high beamforming gains as explicit CSI feedback and as low overhead as implicit CSI feedback.* EliMO significantly reduces computation, communication, and energy overhead for STAs without sacrificing beamforming gains. Fig. 6.2b shows the procedure of two-way channel estimation. The AP sends the training signal X to the STA, and the STA sends the received signal Y_d , in a amplify-and-forward way, back to the AP. The received signal of Y_d at the AP is Y_f , and the AP estimates the two-way channel based on X and Y_f . The STA does not need to demodulate Y_d . Besides, the STA does not need to calculate when and how to send CSI feedback. Thus the computation overhead of the STA is significantly reduced. Moreover, the STA does not need to send CSI back to the AP, so the communication overhead of sending CSI packets for the STA is completely eliminated. The only extra overhead of EliMO compared with implicit CSI feedback for the STA is sending Y_d , which is only $8\mu s$. Finally, the energy consumption of sending CSI packets for the STA is also eliminated. In the following we show SNR and overhead analysis to show that two-way channel estimation is able provide high SNR with low overhead.

6.3.2 SNR Analysis

Since the transmit signal after using beamforming is QX , the effective MIMO channel is $H_{eff} = HQ$ [46]. In practical MIMO systems, there is always a time delay between when downlink CSI is measured and when the measured CSI is used to send the data packet. In this case, the effective CSI is

$$H_{eff} = H_{dd}Q = H_{dd}\hat{H}_d^*(\hat{H}_d\hat{H}_d^*)^{-1}, \quad (6.2)$$

where H_{dd} is the downlink CSI of the data packet, and \hat{H}_d is the measured downlink CSI. Both H_{dd} and \hat{H}_d are baseband-to-baseband channels, which consist of digital baseband and over-the-air channels. For over-the-air channels with multiple paths, the CSI value from the i th transmit antenna to the j th receive antenna at the k th sub-carrier is

$$\check{h}_{ijk} = \sum_n^N a_n e^{-j2\pi d_{ijn}/\lambda_k}, \quad (6.3)$$

where a_n is the attenuation along the n th path, d_{ijn} is the distance between the i th transmit and the j th receive antenna along the n th path, λ_k is the wavelength of the k th sub-carrier, and N is the number of paths [148]. For baseband signal $x(t)$, the corresponding RF signal is $x_{rf}(t) = \text{Re}\{x(t)e^{-j2\pi f_c t}\}$, where f_c is the carrier frequency, and $\text{Re}\{\cdot\}$ returns the real part of the input [61].

The STA uses Minimum Mean Square Error (MMSE) [46, 50, 49] to decode the received signal. The SNR of the k th sub-carrier of the j th spatial stream is $snr_{k,j} = 1/Y_{jj} - 1$, where $Y = (H_{eff,k}^* H_{eff,k} + I_S)^{-1}$, $H_{eff,k}$ is the effective CSI of the k th sub-carrier, and I_S is an $S \times S$ identity matrix with $S = \min(N_t, N_r)$ as the maximum number of streams supported by the MIMO channel. The beamforming errors between H_{dd} and \hat{H}_d influences the receiving SNR: $snr = \text{dB}(\sum snr_{k,j}/\sqrt{S})$, where \sqrt{S} is the scaling factor [46, 50].

Fig. 6.3a shows the Cumulative Distribution Function (CDF) of SNR using implicit/explicit CSI feedback and two-way channel estimation. The initial distance between

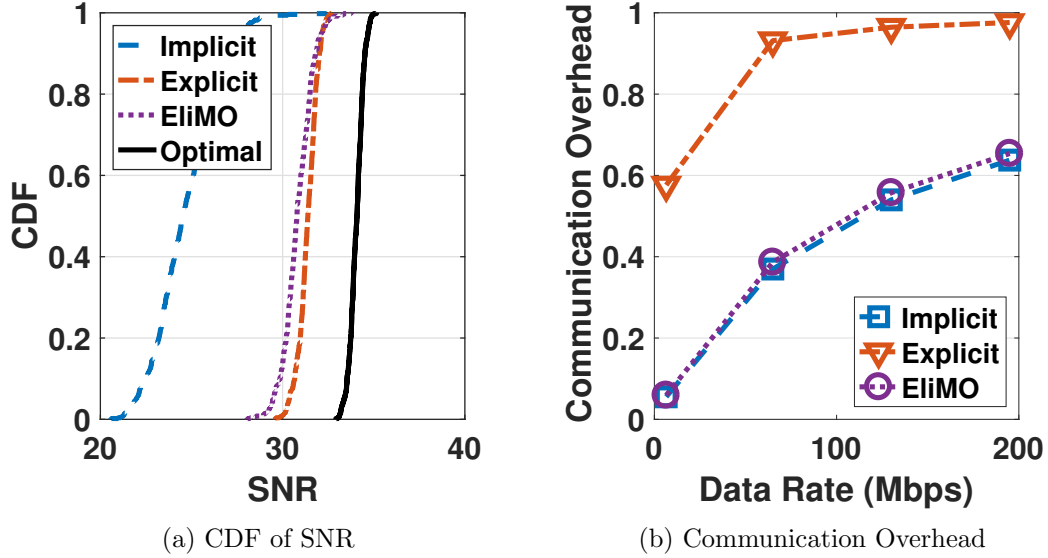


Figure 6.3: SNR and communication overhead analysis. EliMO provides as high SNR as explicit CSI feedback and as low overhead as implicit CSI feedback.

the AP and STA is 5 meters, and the STA moves away from the AP at the speed of 1.2 meters/second. The size of the CSI packet is $3 * 3 * 52 * 32 / 8 = 1,872$ bytes for a 20MHz WiFi channel with 3/3 transmitting/receiving antennas, 52 sub-carriers, and 32 bits of data for each CSI value. The size of the data packet is 1,500 bytes. EliMO has 7dB higher SNR than implicit CSI feedback, and only 0.8dB lower SNR than explicit CSI feedback. SNR analysis results in Fig. 6.3a demonstrate that *EliMO provides as high SNR as explicit CSI feedback*. This is validated by CSI traces from real-world experiments, which will be shown later in §6.5.

6.3.3 Overhead Analysis

For each data packet, control packets are needed for MAC-layer coordinations and beamforming matrix calculations, as shown in Fig. 6.1. Sending and receiving these control packets introduce computation and communication overhead for the STA. The

communication overhead is defined as

$$\tau = \frac{t_{control}}{t_{control} + t_{data}}, \quad (6.4)$$

where $t_{control}$ is the transmission time for control frames, and t_{data} is for data frames. Control frames are always transmitted using the lowest data rate, while data frames can use higher data rates. For a certain CSI feedback scheme, $t_{control}$ is relatively stable. When the data rate for data frames is much higher than that of control frames, which is the common case for 802.11n/ac, t_{data} is much smaller than $t_{control}$. In this case, the communication overhead is extremely high.

Fig. 6.3b shows the results of communication overhead of implicit/explicit CSI feedback and EliMO. The channel width is 20MHz, and the data rate for data frames is in the range of 6.5-195Mbps for up to 3/3 transmitting/receiving antennas [61]. The communication overhead of EliMO is comparable as that of implicit CSI feedback, and is only 40% to 90% lower than that of explicit CSI feedback. Fig. 6.3b demonstrates that *EliMO introduces as low overhead as implicit CSI feedback.*

To sum up, FTF has comparable SNR as explicit CSI feedback and much higher SNR than implicit CSI feedback, as shown in Fig. 6.3a. The communication overhead of FTF is similar to that of implicit CSI feedback, and it is much lower than that of explicit CSI feedback, as shown in Fig. 6.3b.

6.4 EliMO Protocol Design

This section presents the EliMO protocol including frame format, two-way channel estimation, and MAC-layer operations. There are two challenges for EliMO to get high beamforming gains: (1) the AP needs to accurately estimate the downlink CSI in the presence of two-way channel propagations and interferences. (2) the AP needs to determine whether the measured downlink CSI is stale and when to request feedback training.

6.4.1 Frame Format

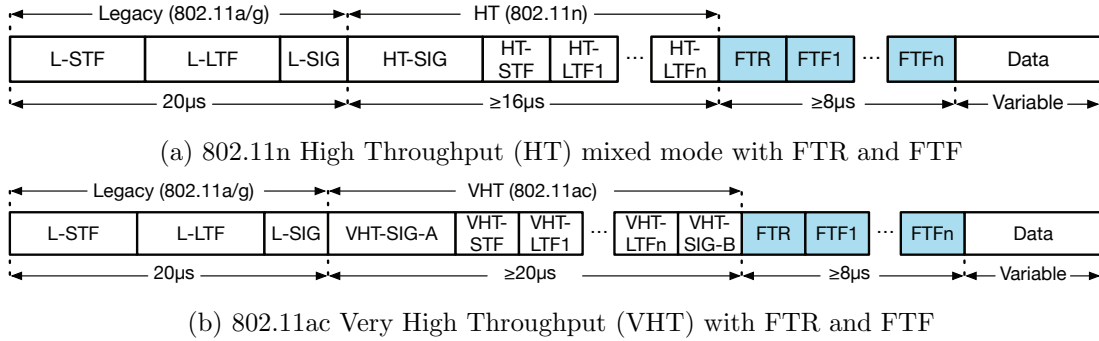


Figure 6.4: Frame format of 802.11n/ac packets with FTR and FTF.

EliMO reuses the frame format of 802.11n/ac packet headers. Fig. 6.4 shows the frame format of EliMO for 802.11n mixed mode and 802.11ac packets. Two new fields, i.e., Feedback Training Request/Response (FTR) and Feedback Training Field (FTF), are inserted after 802.11n/ac packet headers. FTR indicates whether feedback training is requested or not and whether FTF is sent back or not. FTR is inserted right after the 802.11n/ac preamble. If the request field of FTR is 1, the AP sets the response field to 0, which means there is no FTF sent after FTR. When the STA receives the request of feedback training, it sets the response field of FTR to 1 and sends FTF following FTR. FTF is in corresponding with each HT/VHT-LTF. The length of FTR and FTF are both 4 μs. If there is only one HT-LTF or VHT-LTF, the length of FTR plus FTF is 8 μs. Comparing with implicit CSI feedback that typically has 150 μs of control overhead, EliMO introduces only 8 μs extra overhead.

6.4.2 Two-Way Channel Estimation

The AP estimates the downlink CSI using the received signal of FTF, i.e., the received signal of downlink LTF that goes through both the downlink and uplink MIMO channel. The AP needs to accurately estimate the downlink CSI in the presence of two-way channel propagations and interferences. To address this issue, the STA amplifies the received signal

of downlink LTF and sends duplicated FTFs back to the AP.

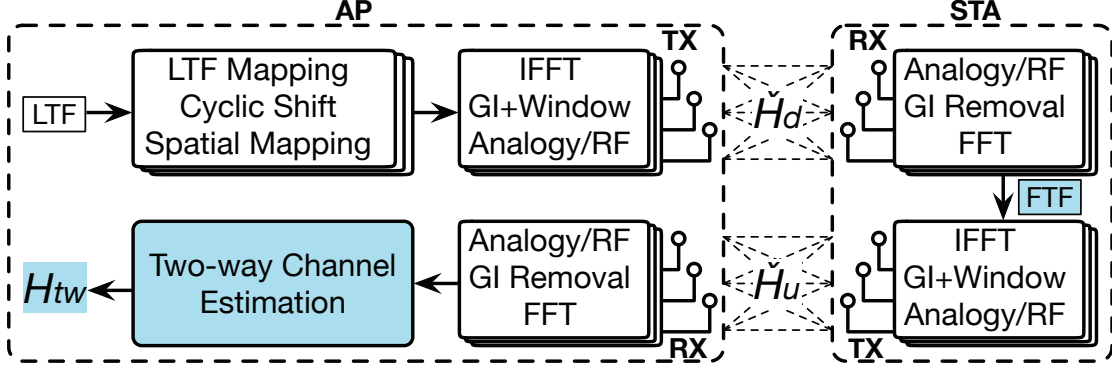


Figure 6.5: Block diagram of two-way channel estimation using FTF.

Fig. 6.5 shows the block diagram of two-way channel estimation. White blocks are components in existing 802.11n/ac systems, and blue blocks are added in EliMO. The AP sends LTF to the STA, which performs FFT on the received signal. The STA performs analogy/RF, Guard Interval (GI) removal, and FFT to include the impact of digital baseband of both the AP and STA. Since the STA does not need to demodulate the received signal, it has lower computation overhead. The STA amplifies the received signal and sends it back to the AP in FTF. The AP estimates two-way channel, i.e., $H_{tw} := H_u H_d$, by the received signal and the original LTF. The AP also estimate the uplink CSI \hat{H}_u by the received signal of uplink LTF from the STA. We use the pseudo-inverse of \hat{H}_u and two-way channel H_{tw} to estimate the downlink CSI \hat{H}_d , i.e.,

$$\hat{H}_d = \hat{H}_u^+ H_{tw} = (\hat{H}_u^* \hat{H}_u)^{-1} \hat{H}_u^* H_{tw} = (\hat{H}_u^* \hat{H}_u)^{-1} \hat{H}_u^* H_u H_d, \quad (6.5)$$

where $\hat{H}_u^+ = (\hat{H}_u^* \hat{H}_u)^{-1} \hat{H}_u^*$ is the pseudo-inverse of \hat{H}_u .

The estimation accuracy of \hat{H}_d is impacted by two-way channel propagations and interferences. The received signal of FTF at the AP is

$$Y_f = H_u Y_d + N_u = H_u (H_d X + N_d) + N_u, \quad (6.6)$$

where N_d and N_u are downlink and uplink noise signals, respectively. The power of Y_f is impacted by two-way channel propagations, so the CSI estimation accuracy could be significantly influenced by the power of N_u . The STA sends the amplified signal αY_d , instead of Y_d , to the AP to improve estimation accuracy. The amplify factor α is constrained by $E[|\alpha Y_d|^2] \leq P_i$, where P_i is the power constraint for the i th transmit antenna of the STA [69]. To reduce the impact of two-way channel interferences, the AP sends duplicated LTFs to the STA, and correspondingly the STA also sends duplicated FTFs to the AP. The received signals for the k th sub-carrier at the AP are $Y_f(k, 1)$ and $Y_f(k, 2)$. The AP first estimates uplink noise by $\hat{N}_u = \sum_{k=1}^{N_s} |(Y_f(k, 1) - Y_f(k, 2))(Y_f(k, 1) - Y_f(k, 2))^*|$ [195]. The AP estimates two-way channel H_{tw} and uplink channel \hat{H}_u based on \hat{N}_u , and finally estimates downlink channel \hat{H}_d using \hat{H}_u and H_{tw} .

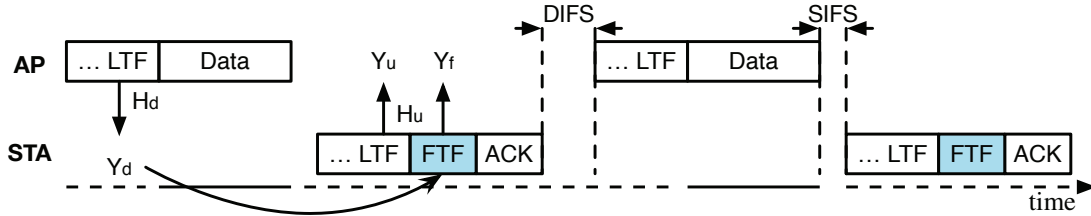


Figure 6.6: MAC-layer operations of EliMO.

6.4.3 MAC-Layer Operations

Timeline of MAC-layer operations of EliMO is shown in Fig. 6.6. The AP sends downlink LTF to the STA, and the STA puts the received downlink LTF signal Y_d in FTF. The STA sends FTF, along with uplink LTF, back to the AP. The received signal of uplink LTF is Y_u , and the received signal of FTF is Y_f . The AP estimates uplink CSI \hat{H}_u using Y_u and two-way CSI using Y_f .

The beamforming performance of EliMO is influenced by the difference between \hat{H}_d and H_{dd} . The AP needs to determine when to request feedback training to reduce beamforming errors. EliMO addresses this issue by sending feedback request when the AP detects that

the previous measured downlink CSI is stale. If \hat{H}_d is stale, the AP needs to send Null Data Packet (NDP) with FTR to measure the current downlink CSI. The AP uses two metrics, CSI similarity and estimation delay, to determine whether \hat{H}_d is stale or not. CSI similarity is calculated by

$$\rho = \frac{\sum_{k=1}^{N_s} (h(k, 1) - \bar{h}_1)(h(k, 2) - \bar{h}_2)}{\sqrt{\sum_{k=1}^{N_s} (h(k, 1) - \bar{h}_1)^2} \sqrt{\sum_{k=1}^{N_s} (h(k, 2) - \bar{h}_2)^2}}, \quad (6.7)$$

where $h(k, 1)$ and $h(k, 2)$ are the CSI magnitude of the k th sub-carrier, and \bar{h}_1 and \bar{h}_2 are the average CSI magnitude across N_s sub-carriers of two CSI measurements [46, 143]. When CSI similarity of either downlink or uplink CSI is larger than the threshold Thr_ρ , the AP sends NDP with FTR to the STA to measure the current downlink CSI. Based on experiment measurements, which will be shown in the next section, we find that $Thr_\rho = 0.98$ is able to distinguish whether the STA is moving or not. This is also in consistent with measurement results in [143]. Thus we use $Thr_\rho = 0.98$ as the CSI similarity threshold. Estimation delay δ is the time interval between when the previous downlink CSI is estimated and when the next data packet is transmitted. The AP also sends NDP when the estimation delay is larger than the threshold Thr_δ . Based on experiment results in both static and mobile scenarios, we choose $Thr_\delta = 100\text{ms}$ as the threshold of estimation delay. Note that all calculations of detecting whether \hat{H}_d is stale or not are done by the AP. No extra computation overhead is introduced for the STA.

Fig. 6.7 and 6.8 show the flow chart of EliMO for the AP and STA. For each data packet to be sent, the AP checks whether the previous downlink CSI \hat{H}_d is stale or not based on CSI similarity ρ and estimation delay δ . If \hat{H}_d is stale, the AP sends NDP with FTR to the STA to measure the current downlink CSI. If \hat{H}_d is not stale, the AP calculates the beamforming matrix based on \hat{H}_d and sends the data packet to the STA. In the packet header of the data packet, the AP sets the request field of FTR to 1. This is for estimating the downlink CSI for the next data packet. For each received packet, the AP estimates the uplink CSI \hat{H}_u by uplink LTF and two-way CSI H_{tw} by FTF. The downlink CSI \hat{H}_d

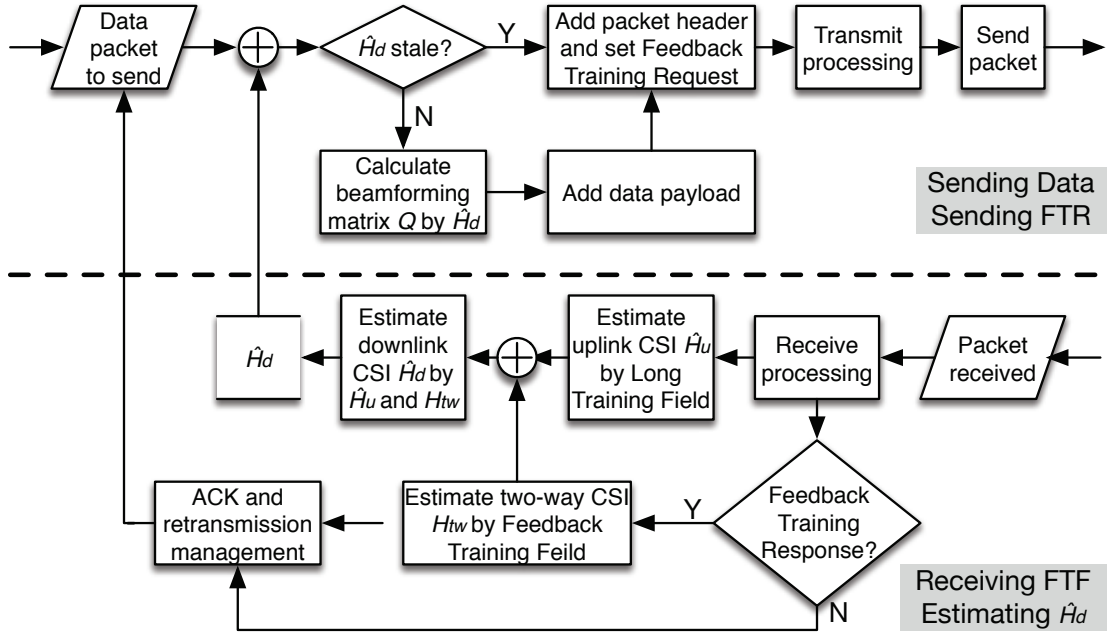


Figure 6.7: Flow chart of MAC-layer operations of the AP side.

is calculated by \hat{H}_u and H_{tw} . At the STA side, each received packet is checked whether it contains data payload or not. If the received packet has no data payload, demodulation is not needed. If the packet has data payload, the STA demodulates and decodes the received signal to get the data bits. The STA checks correctness of the data packet and adds ACK payload to the packet to be sent to the AP. If feedback training is requested, the STA gets the received signal of downlink LTF, puts it in FTF, and sets the response field of FTR to 1. Finally, the STA adds packet header with FTR and FTF, and sends the packet, either with or without ACK payload, to the AP.

6.5 Evaluation

This section presents evaluation results, including throughput and energy consumption, of EliMO by experiment measurements in both static and mobile scenarios.

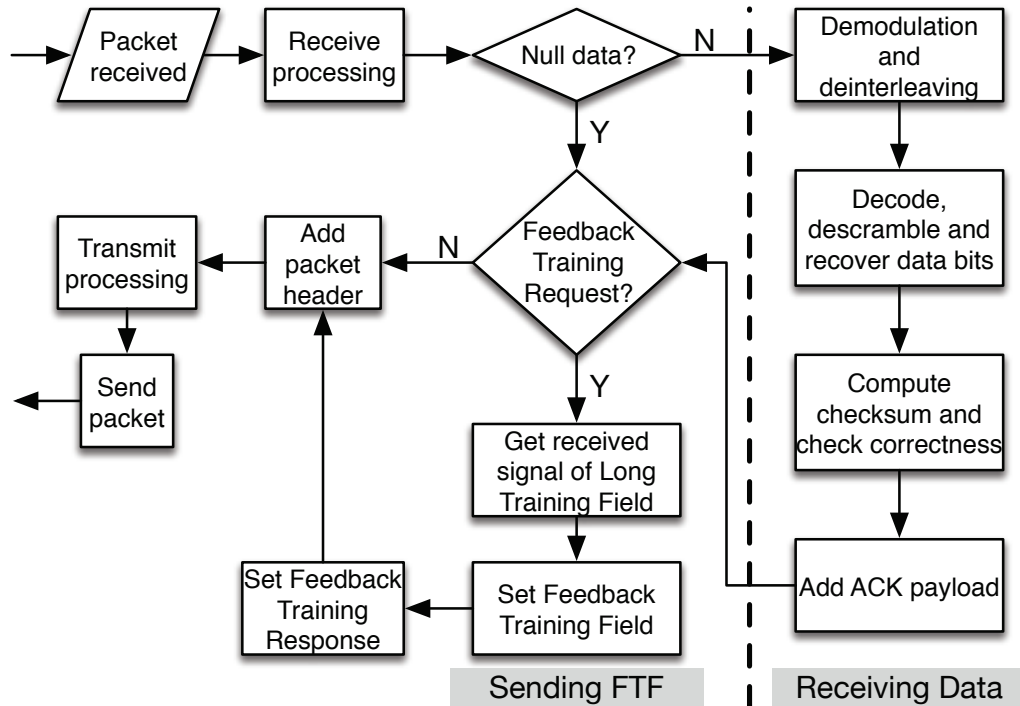


Figure 6.8: Flow chart of MAC-layer operations of the STA side.

6.5.1 Experiment Setup

We conduct experiment measurements in indoor environments for both static and mobile scenarios. The AP is static, and the STA is either static or moving at the speed of about 1.2m/s. The AP and STA operate at 5GHz, and the channel width is 20MHz. The AP has 3 external antennas, and the STA has 3 internal antennas spaced 2.4 inches apart. The transmitting power of the AP/STA is fixed at 17/15dBm. The AP and STA are two laptops with Intel WiFi Link 5300 installed. Since we cannot program the power signal of the WiFi chipset, we are not able to implement EliMO in real-time. Thus we employ trace-driven evaluation by collecting CSI traces and evaluates EliMO off-line by Matlab implementations using the collected CSI traces.

Downlink and uplink CSI measurements are collected using openrf [76], which is based on 802.11n CSI tool [51]. Note that 802.11n CSI tool only provides CSI values of 30 sub-carriers even though a 20MHz WiFi channel has 52 sub-carriers [61, 62, 34, 114].

Two performance metrics, throughput and energy consumption, are evaluated in different scenarios comparing EliMO with implicit and explicit CSI feedback. The AP uses ZFBF as the transmit beamforming algorithm and the STA uses the MMSE receiving algorithm. The MCS index can be selected from 0 to 23 with the data rate ranging from 6.5 to 195Mbps [61]. We compare EliMO with implicit and explicit CSI feedback. There are two options for explicit CSI feedback: non-compressed, i.e., 1 CSI packet per data packet, and compressed, i.e., 1 CSI packet per 10 data packets.

6.5.2 Experiment Validation

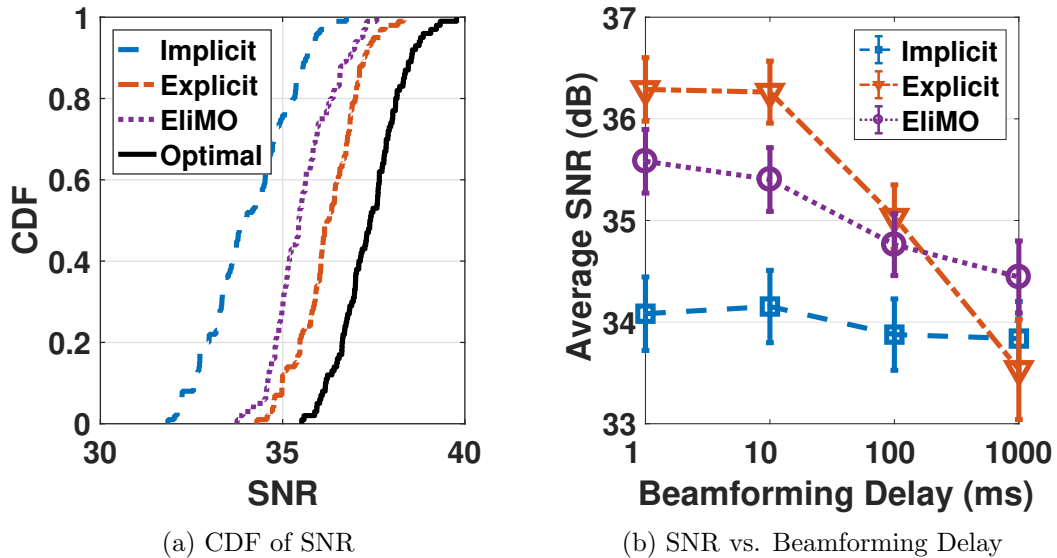


Figure 6.9: Experiment validation of SNR results.

We first validate the effectiveness of two-way channel estimation by experiments in real-world MIMO systems. Fig. 6.9 shows the CDF of SNR of experiment measurements in the mobile scenario. The SNR of EliMO is 1.5dB higher than that of implicit CSI feedback, and 1dB lower than that of explicit CSI feedback, as shown in Fig. 6.9a. EliMO is able to provide comparable SNR as explicit CSI feedback in real-world mobile environments. We also check the impact of beamforming delay, as shown in Fig. 6.9b. The average SNR of implicit CSI feedback does not change much as beamforming delay increases. The reason

is that channel reciprocity has more impact on the accuracy of downlink CSI estimation than mobility. The average SNR of EliMO and explicit CSI feedback decreases when beamforming delay increases. This is because that the difference between the estimated CSI and data packet CSI increases as beamforming delay increases when the STA is moving. For beamforming delay of 1,000ms, the SNR decrease is 1dB for EliMO and 2.8dB for explicit CSI feedback. EliMO has lower SNR decrease since the time delay between uplink CSI and data packet CSI is very small and two-way channel estimation helps reduce the impact of mobility.

6.5.3 Throughput

The effective throughput is calculated by

$$tpt = \frac{\sum_{i=1}^{N'} size(pkt_i)}{t_{control} + t_{data}}, \quad (6.8)$$

where pkt_i is the i th data packet, and N' is the number of received packets. Implicit CSI feedback has low accuracy of downlink CSI estimation, so it provides low beamforming gains. This reduces the number of received packets N' and leads to low throughput for implicit CSI feedback. The transmission time of control frames $t_{control}$ is extremely high, which results in low throughput, for explicit CSI feedback. EliMO provides high throughput by reducing $t_{control}$ significantly while N' is not seriously influenced.

Fig. 6.10a shows the average throughput for data packets of different sizes and data rates. The average throughput of EliMO is 5×, 4×, and 1.7× of that of implicit, non-compressed explicit, and compressed explicit CSI feedback, respectively. The average throughput of the mobile scenario is lower than that of static scenarios for all feedback schemes. In the mobile scenario, EliMO still provides the highest throughput. The average throughput of EliMO is 3.6×/4.5× of that of implicit/explicit CSI feedback. Fig. 6.10b shows the average throughput as the size of data packets changes. For data size of 1,024 bytes, the average throughput of EliMO is 6Mbps, 7Mbps, and 5Mbps higher than that

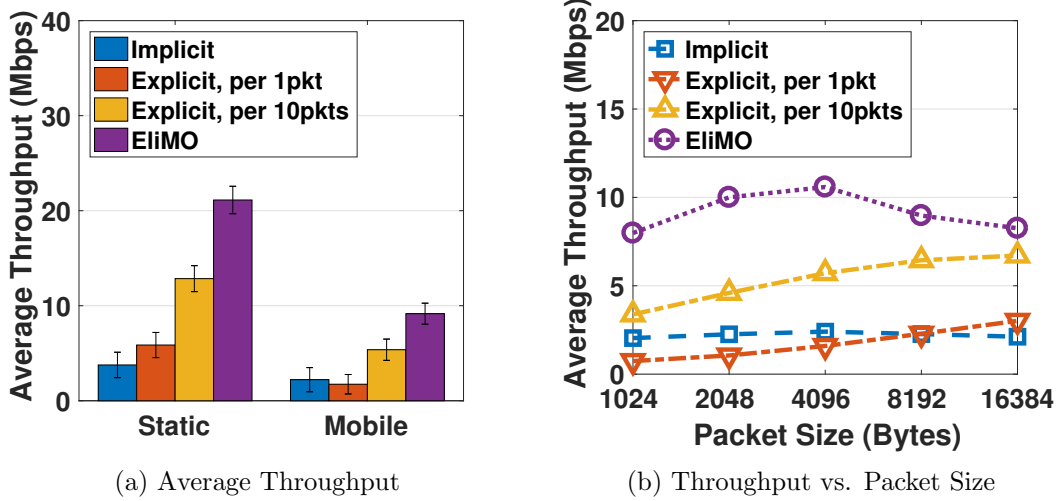


Figure 6.10: Evaluation results of average throughput.

of implicit, non-compressed explicit, and compressed explicit CSI feedback. For data size of 16,384 bytes, EliMO has 6Mbps, 5Mbps, and 1Mbps higher throughput than the other three feedback schemes.

6.5.4 Energy Efficiency

Energy efficiency of the STA is evaluated by energy consumption per data bit

$$eb = \frac{\sum_{i=1}^N (er(0) * size(ctr_i) + et(0) * size(csi_i))}{\sum_{i=1}^{N'} size(pkt_i)} + \frac{\sum_{i=1}^N er(m_i) * size(pkt_i)}{\sum_{i=1}^{N'} size(pkt_i)}, \quad (6.9)$$

where $et(m)$ and $er(m)$ stand for energy consumption per bit for transmitting and receiving, respectively, as using MCS index m [48, 131]. Energy consumption parameters, $et(m_i)$ and $er(m_i)$, for the Intel 5300 WiFi chipset are from [48]. For data packet of $size(pkt_i) = 1,500$ bytes and $m_i = 23$, explicit CSI feedback accounts 80% of the total energy consumption. Besides, $et(m_i)|_{m_i=0}$ for CSI packets is much larger than $er(m_i)|_{m_i \geq 0}$ for data packets [48, 131], so explicit CSI feedback consumes a lot of energy for the STA. For implicit CSI feedback, the number of received packets is smaller than EliMO due to low accuracy of downlink CSI estimation. This reduces the energy efficiency of the STA

for implicit CSI feedback. EliMO remarkably improves the energy efficiency of the STA by eliminating explicit CSI feedback without sacrificing beamforming gains.

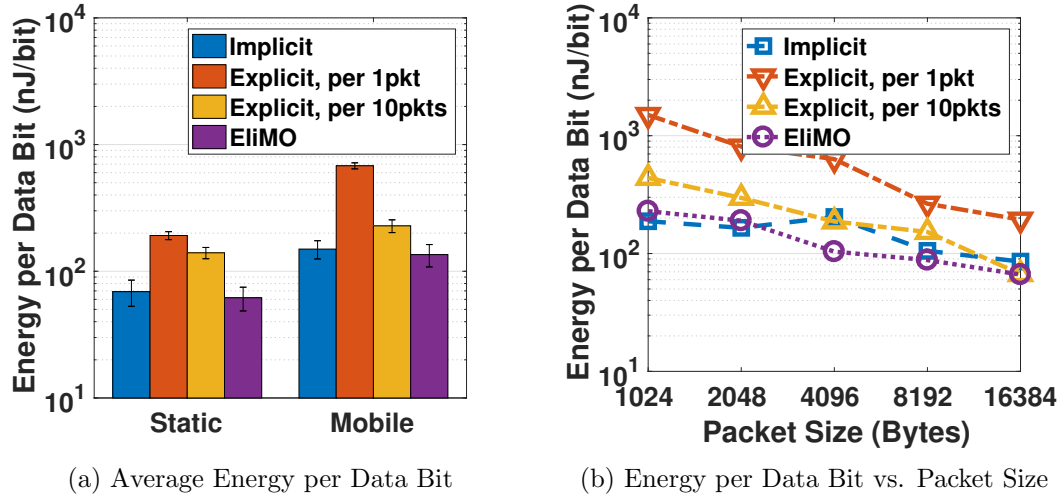


Figure 6.11: Evaluation results of energy consumption.

Energy consumption results are shown in Fig. 6.11. As shown in Fig. 6.11a, EliMO has slightly lower energy consumption as implicit CSI feedback in both static and mobile scenarios. For the static scenario, the average energy consumption of EliMO is only 30% and 50% of that of non-compressed and compressed explicit CSI feedback, respectively. For the mobile scenario, the average energy consumption of EliMO is 17%/57% of that of non-compressed/compressed explicit CSI feedback. Fig. 6.11b shows average energy consumption in terms of the size of data packets. For data packets of less than 2,048 bytes, EliMO consumes slightly higher energy than implicit CSI feedback. As packet size increases, energy consumption of EliMO is lower than that of implicit CSI feedback. For packet size of 16,384 bytes, EliMO has comparable energy consumption as compressed explicit CSI feedback. EliMO consumes much less energy than both non-compressed and compressed explicit CSI feedback when packet size is less than 16,384 bytes.

6.6 Chapter Summary

In this chapter, we show that implicit CSI feedback has low beamforming gains and explicit CSI feedback has high computation and communication overhead. We propose EliMO to eliminate CSI feedback without sacrificing beamforming gains. We propose Feedback Training Field and two-way channel estimation to enable the AP to accurately estimate downlink CSI without explicit CSI feedback. Based on both theoretical analysis and experiment measurements, EliMO provides as low overhead as implicit CSI feedback and as high SNR as explicit CSI feedback. Experiment evaluation results show that EliMO provides much higher throughput and lower energy consumption for the STA than implicit and explicit CSI feedback. EliMO significantly reduces computation and communication costs of measuring and sending CSI feedback for smart devices, like smartphones, smartwatches, and wireless drones.

Chapter 7

Conclusions and Future Works

7.1 Conclusions

This dissertation presents how CSI can be used for improving WiFi sensing and networking. For WiFi sensing, the dissertation first presents a survey of signal processing techniques, algorithms, applications, performance results, challenges, and future trends of WiFi sensing with CSI. It gives a summary of the advantages and limitations of different algorithms for different WiFi sensing applications. It presents three key challenges for WiFi sensing and three future trends for improving existing WiFi sensing applications and enabling new WiFi sensing opportunities. Second, the dissertation presents SignFi for recognizing 276 sign gestures with CSI and CNNs. It proposes a signal processing technique for removing CSI phase offsets and a 9 layer CNN for recognizing 276 sign gestures with high accuracy and low testing cost. Third, the dissertation presents a deep learning solution with neural networks and reinforcement learning for person and location independent activity recognition with WiFi. The proposed solution uses a 2D CNN as the recognition algorithm for learning person and location independent features, a 1D CNN as the state machine for learning time dependency information, and an RNN with LSTM as the reinforcement learning agent for neural architecture search.

For improving WiFi networking, the dissertation first presents RoFi to reduce CSI

feedback based on the mobility status of WiFi receivers. It shows that WiFi has different CSI feedback requirements when the receiver is in different mobility status, i.e., static, moving, and rotating. It demonstrates that existing approaches cannot distinguish between moving and rotating and proposes a new metric to recognize whether the receiver is rotating or not. RoFi sends CSI feedback only when it is necessary based on the mobility status of the receiver. It reduces about 20% CSI feedback overhead compared with existing methods in different mobility scenarios. RoFi and other CSI feedback compression approaches reduces overhead, but they still require WiFi receiver measure CSI, calculate when to send CSI and how much CSI to send, and send CSI back to the transmitter. This dissertation presents EliMO with two-way channel estimation to eliminate CSI feedback without sacrificing beamforming gains. EliMO achieves as high SNR as explicit CSI feedback and as low overhead as implicit CSI feedback. It significantly reduces the computation and communication overhead for WiFi receivers.

7.2 Future Works

For SignFi, CSI measurements are manually segmented for each sign gesture, so it only supports word-level sign language recognition. For sentence-level sign language recognition, CSI measurements should be automatically segmented which introduces many new challenges. RNN with LSTM could help for automatic sentence-level sign language recognition. SignFi is tested to be robust for two environments and five users. There are many other factors that may influence the recognition performance. For example, the distance between the person and the AP/STA could be different. The direction and orientation of the person with respect to the AP/STA could also change. There could be multiple persons or other moving objects around. The person or other objects could block the direct path from the AP to STA. More CSI traces could be collected in different environments and scenarios for performance evaluation and tests considering these factors.

Currently EliMO is evaluated by CSI measurement traces and off-line Matlab imple-

mentations. EliMO could be implemented on real-world MIMO systems and be evaluated in real-time. EliMO could be adapted to be compatible with multi-user MIMO systems in the future. EliMO can also be used by other applications, such as motion tracking [144], activity recognition [28], and localization [75, 151, 200], using off-the-shelf WiFi chipsets. For example, existing CSI-based localization methods require extensive CSI measurements from multiple APs [75] or across multiple channels [151, 200], which could introduce high overhead for WiFi receivers. EliMO can help reduce both computation and communication costs for WiFi receivers, like smartwatches and drones, for CSI-based localization approaches. Finally, WiFi sensing results, along with machine learning and reinforcement learning, can be used to improve the performance and efficiency of WiFi networking.

Bibliography

- [1] HEBA ABDELNASSER, KHALED A. HARRAS, AND MOUSTAFA YOUSSEF. Ubibreathe: A ubiquitous non-invasive wifi-based breathing estimator. In *Proceedings of the 16th ACM International Symposium on Mobile Ad Hoc Networking and Computing*, MobiHoc '15, pages 277–286, 2015.
- [2] HEBA ABDELNASSER, MOUSTAFA YOUSSEF, AND KHALED A. HARRAS. Wigest: A ubiquitous wifi-based gesture recognition system. In *2015 IEEE Conference on Computer Communications (INFOCOM)*, pages 1472–1480, April 2015.
- [3] FADEL ADIB AND DINA KATABI. See through walls with wifi! In *Proceedings of the ACM SIGCOMM 2013 Conference on SIGCOMM*, SIGCOMM '13, pages 75–86, 2013.
- [4] KAMRAN ALI, ALEX X. LIU, WEI WANG, AND MUHAMMAD SHAHZAD. Keystroke recognition using wifi signals. In *Proceedings of the 21st Annual International Conference on Mobile Computing and Networking*, MobiCom '15, pages 90–102. ACM, 2015.
- [5] KAMRAN ALI, ALEX X. LIU, WEI WANG, AND MUHAMMAD SHAHZAD. Recognizing keystrokes using wifi devices. *IEEE Journal on Selected Areas in Communications*, 35(5):1175–1190, May 2017.
- [6] SHEHERYAR ARSHAD, CHUNHAI FENG, YONGHE LIU, YUPENG HU, RUIYUN YU, SIWANG ZHOU, AND HENG LI. Wi-Chase: A WiFi based Human Activity Recognition System for Sensorless Environments. In *2017 IEEE 18th International Symposium on A World of Wireless, Mobile and Multimedia Networks (WoWMoM)*, pages 1–6, June 2017.
- [7] IBRAHIM ETHEM BAGCI, UTZ ROEDIG, IVAN MARTINOVIC, MATTHIAS SCHULZ, AND MATTHIAS HOLLICK. Using channel state information for tamper detection in the inter-

- net of things. In *Proceedings of the 31st Annual Computer Security Applications Conference, ACSAC 2015*, pages 131–140. ACM, 2015.
- [8] ARIJIT BANERJEE, DUSTIN MAAS, MAURIZIO BOCCA, NEAL PATWARI, AND SNEHA KASERA. Violating privacy through walls by passive monitoring of radio windows. In *Proceedings of the 2014 ACM Conference on Security and Privacy in Wireless & Mobile Networks, WiSec '14*, pages 69–80, 2014.
- [9] MARCO V. BARBERA, ALESSANDRO EPASTO, ALESSANDRO MEI, VASILE C. PERTA, AND JULINDA STEFA. Signals from the crowd: Uncovering social relationships through smartphone probes. In *Proceedings of the 2013 Conference on Internet Measurement Conference, IMC '13*, pages 265–276. ACM, 2013.
- [10] ELI BENDERSKY. Depthwise separable convolutions for machine learning, April 2018. <https://eli.thegreenplace.net/2018/depthwise-separable-convolutions-for-machine-learning/>.
- [11] SEONGHO BYEON, KANGJIN YOON, OKHWAN LEE, SUNGHYUN CHOI, WOONSUN CHO, AND SEUNGSEOK OH. MoFA: Mobility-aware frame aggregation in Wi-Fi. In *ACM CoNEXT*, 2014.
- [12] NAOMI K. CASELLI, ZED SEVCIKOVA SEHYR, ARIEL M. COHEN-GOLDBERG, AND KAREN EMMOREY. Asl-lex: A lexical database of american sign language. *Behavior Research Methods*, 49(2):784–801, April 2017.
- [13] LIWEI CHAN, RONG-HAO LIANG, MING-CHANG TSAI, KAI-YIN CHENG, CHAO-HUAI SU, MIKE Y. CHEN, WEN-HUANG CHENG, AND BING-YU CHEN. Fingerpad: Private and subtle interaction using fingertips. In *Proceedings of the 26th Annual ACM Symposium on User Interface Software and Technology, UIST '13*, pages 255–260, 2013.
- [14] KE-YU CHEN, KENT LYONS, SEAN WHITE, AND SHWETAK PATEL. utrack: 3d input using two magnetic sensors. In *Proceedings of the 26th Annual ACM Symposium on User Interface Software and Technology, UIST '13*, pages 237–244, 2013.

- [15] KE-YU CHEN, SHWETAK N. PATEL, AND SEAN KELLER. Finexus: Tracking precise motions of multiple fingertips using magnetic sensing. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*, CHI '16, pages 1504–1514, 2016.
- [16] YUANYING CHEN, WEI DONG, YI GAO, XUE LIU, AND TAO GU. Rapid: A multimodal and device-free approach using noise estimation for robust person identification. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.*, 1(3):41:1–41:27, September 2017.
- [17] LINSONG CHENG AND JILIANG WANG. How can i guard my ap?: Non-intrusive user identification for mobile devices using wifi signals. In *Proceedings of the 17th ACM International Symposium on Mobile Ad Hoc Networking and Computing*, MobiHoc '16, pages 91–100, 2016.
- [18] FRANÇOIS CHOLLET. Xception: Deep learning with depthwise separable convolutions. *arXiv:1610.02357*, 2016.
- [19] FRANCOIS CHOLLET. *Deep Learning with Python*. Manning Publications Co., Greenwich, CT, USA, 1st edition, 2017.
- [20] YOHAN CHON, SUYEON KIM, SEUNGWOO LEE, DONGWON KIM, YUNGEUN KIM, AND HOJUNG CHA. Sensing wifi packets in the air: Practicality and implications in urban mobility monitoring. In *Proceedings of the 2014 ACM International Joint Conference on Pervasive and Ubiquitous Computing*, UbiComp '14, pages 189–200, 2014.
- [21] CHING-HUA CHUAN, ERIC REGINA, AND CAROLINE GUARDINO. American sign language recognition using leap motion sensor. In *Proceedings of the 2014 13th International Conference on Machine Learning and Applications*, ICMLA '14, pages 541–544, 2014.
- [22] R. CREPALDI, JEONGKEUN LEE, R. ETKIN, SUNG-JU LEE, AND R. KRAVETS. CSI-SF: Estimating wireless channel state using CSI sampling & fusion. In *IEEE INFOCOM*, 2012.
- [23] MAURO DE SANCTIS, ERNESTINA CIANCA, SIMONE DI DOMENICO, DANIELE PROVENZANI, GIUSEPPE BIANCHI, AND MARINA RUGGIERI. Wibecam: Device free human activity recognition through wifi beacon-enabled camera. In *Proceedings of the 2Nd Workshop on Workshop on Physical Analytics*, WPA '15, pages 7–12. ACM, 2015.
- [24] SIMONE DI DOMENICO, MAURO DE SANCTIS, ERNESTINA CIANCA, AND GIUSEPPE BIANCHI. A trained-once crowd counting method using differential wifi channel state infor-

- mation. In *Proceedings of the 3rd International on Workshop on Physical Analytics, WPA '16*, pages 37–42. ACM, 2016.
- [25] SHIHONG DUAN, TIANQING YU, AND JIE HE. Widriver: Driver activity recognition system based on wifi csi. *International Journal of Wireless Information Networks*, Feb 2018.
- [26] BIYI FANG, JILLIAN CO, AND MI ZHANG. DeepASL: Enabling Ubiquitous and Non-Intrusive Word and Sentence-Level Sign Language Translation. In *Proceedings of the 15th ACM Conference on Embedded Networked Sensor Systems, SenSys '17*, 2017.
- [27] BIYI FANG, NICHOLAS D. LANE, MI ZHANG, AIDAN BORAN, AND FAHIM KAWSAR. 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, MobiSys '16*, pages 97–110. ACM, 2016.
- [28] BIYI FANG, NICHOLAS D. LANE, MI ZHANG, AIDAN BORAN, AND FAHIM KAWSAR. BodyScan: Enabling radio-based sensing on wearable devices for contactless activity and vital sign monitoring. In *ACM MobiSys*, 2016.
- [29] BIYI FANG, NICHOLAS D. LANE, MI ZHANG, AND FAHIM KAWSAR. Headscan: A wearable system for radio-based sensing of head and mouth-related activities. In *2016 15th ACM/IEEE International Conference on Information Processing in Sensor Networks (IPSN)*, pages 1–12, April 2016.
- [30] CHUNHAI FENG, SHEHERYAR ARSHAD, AND YONGHE LIU. Mais: Multiple activity identification system using channel state information of wifi signals. In *Wireless Algorithms, Systems, and Applications*, pages 419–432. Springer International Publishing, 2017.
- [31] MAKIKO FUNASAKA, YU ISHIKAWA, MASAMI TAKATA, AND KAZUKI JOE. Sign language recognition using leap motion controller. In *Proceedings of the International Conference on Parallel and Distributed Processing Techniques and Applications (PDPTA)*, 2015.
- [32] CHUHAN GAO, YILONG LI, AND XINYU ZHANG. Livetag: Sensing human-object interaction through passive chipless wifi tags. In *15th USENIX Symposium on Networked Systems Design and Implementation (NSDI 18)*, pages 533–546, 2018.

- [33] QINHUA GAO, JIE WANG, XIAORUI MA, XUEYAN FENG, AND HONGYU WANG. Csi-based device-free wireless localization and activity recognition using radio image features. *IEEE Transactions on Vehicular Technology*, 66(11):10346–10356, Nov 2017.
- [34] MATTHEW GAST. *802.11ac: A survival guide*. O’Reilly Media, Inc., 2013.
- [35] DOMENICO GIUSTINIANO, THEODOROS BOURCHAS, MACIEJ BEDNAREK, AND VINCENT LENDERS. Deep inspection of the noise in WiFi time-of-flight echo techniques. In *ACM MSWiM*, 2015.
- [36] JON GJENGSET, JIE XIONG, GRAEME MCPHILLIPS, AND KYLE JAMIESON. Phaser: Enabling phased array signal processing on commodity wifi access points. In *Proceedings of the 20th Annual International Conference on Mobile Computing and Networking, MobiCom ’14*, pages 153–164, 2014.
- [37] DAVID GOLDBERG, DENNIS LOONEY, AND NATALIA LUSIN. Enrollments in languages other than english in united states institutions of higher education, fall 2013. In *Modern Language Association*, 2015.
- [38] LIANGYI GONG, WU YANG, DAPENG MAN, GUOZHONG DONG, MIAO YU, AND JIGUANG LV. Wifi-based real-time calibration-free passive human motion detection. *Sensors*, 15(12):32213–32229, 2015.
- [39] LIANGYI GONG, WU YANG, ZIMU ZHOU, DAPENG MAN, HAIBIN CAI, XIANCUN ZHOU, AND ZHENG YANG. An adaptive wireless passive human detection via fine-grained physical layer information. *Ad Hoc Networks*, 38:38 – 50, 2016.
- [40] IAN GOODFELLOW, YOSHUA BENGIO, AND AARON COURVILLE. *Deep Learning*. MIT Press, 2016. <http://www.deeplearningbook.org>.
- [41] IAN GOODFELLOW, JEAN POUGET-ABADIE, MEHDI MIRZA, BING XU, DAVID WARDEFARLEY, SHERJIL OZAIR, AARON COURVILLE, AND YOSHUA BENGIO. Generative Adversarial Networks. *arXiv:1406.2661*, June 2014.
- [42] YU GU, JINHAI ZHAN, YUSHENG JI, JIE LI, FUJI REN, AND SHANGBING GAO. Mosense: An rf-based motion detection system via off-the-shelf wifi devices. *IEEE Internet of Things Journal*, 4(6):2326–2341, Dec 2017.

- [43] LINLIN GUO, LEI WANG, JIALIN LIU, WEI ZHOU, AND BINGXIAN LU. Huac: Human activity recognition using crowdsourced wifi signals and skeleton data. *Wireless Communications and Mobile Computing*, 2018.
- [44] XIAONAN GUO, BO LIU, CONG SHI, HONGBO LIU, YINGYING CHEN, AND MOOI CHOO CHUAH. Wifi-enabled smart human dynamics monitoring. In *Proceedings of the 15th ACM Conference on Embedded Network Sensor Systems, SenSys '17*, pages 16:1–16:13, 2017.
- [45] SIDHANT GUPTA, DANIEL MORRIS, SHWETAK PATEL, AND DESNEY TAN. Soundwave: Using the doppler effect to sense gestures. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, CHI '12*, pages 1911–1914, 2012.
- [46] DANIEL HALPERIN. *Simplifying the Configuration of 802.11 Wireless Networks with Effective SNR*. PhD thesis, University of Washington, 2012.
- [47] DANIEL HALPERIN. *Simplifying the Configuration of 802.11 Wireless Networks with Effective SNR*. PhD thesis, University of Washington, Seattle, WA, USA, 2013.
- [48] DANIEL HALPERIN, BEN GREENSTEIN, ANMOL SHETH, AND DAVID WETHERALL. Demystifying 802.11n power consumption. In *USENIX HotPower*, 2010.
- [49] DANIEL HALPERIN, WENJUN HU, ANMOL SHETH, AND DAVID WETHERALL. 802.11 with multiple antennas for dummies. *ACM SIGCOMM Comput. Commun. Rev.*, 40(1):19–25, 2010.
- [50] DANIEL HALPERIN, WENJUN HU, ANMOL SHETH, AND DAVID WETHERALL. Predictable 802.11 packet delivery from wireless channel measurements. In *ACM SIGCOMM*, 2010.
- [51] DANIEL HALPERIN, WENJUN HU, ANMOL SHETH, AND DAVID WETHERALL. Tool release: Gathering 802.11n traces with channel state information. *SIGCOMM Comput. Commun. Rev.*, 41(1):53–53, January 2011.
- [52] CHUNMEI HAN, KAISHUN WU, YUXI WANG, AND LIONEL M. NI. Wifall: Device-free fall detection by wireless networks. In *2014 IEEE Conference on Computer Communications (INFOCOM)*, pages 271–279, April 2014.

- [53] WENFENG HE, KAISHUN WU, YONGPAN ZOU, AND ZHONG MING. Wig: Wifi-based gesture recognition system. In *2015 24th International Conference on Computer Communication and Networks (ICCCN)*, pages 1–7, Aug 2015.
- [54] FENG HONG, XIANG WANG, YANNI YANG, YUAN ZONG, YULIANG ZHANG, AND ZHONGWEN GUO. Wfid: Passive device-free human identification using wifi signal. In *Proceedings of the 13th International Conference on Mobile and Ubiquitous Systems: Computing, Networking and Services*, MOBIQUITOUS 2016, pages 47–56. ACM, 2016.
- [55] DONNY HUANG, RAJALAKSHMI NANDAKUMAR, AND SHYAMNATH GOLLAKOTA. Feasibility and limits of wi-fi imaging. In *Proceedings of the 12th ACM Conference on Embedded Network Sensor Systems*, SenSys '14, pages 266–279, 2014.
- [56] JIE HUANG, WENGANG ZHOU, HOUQIANG LI, AND WEIPING LI. Sign language recognition using 3d convolutional neural networks. In *2015 IEEE International Conference on Multimedia and Expo (ICME)*, pages 1–6, June 2015.
- [57] K. HUANG, R. W. HEATH JR., AND J. G. ANDREWS. Limited feedback beamforming over temporally-correlated channels. *IEEE Transactions on Signal Processing*, 57(5):1959–1975, 2009.
- [58] NATHANIEL HUSTED AND STEVEN MYERS. Mobile location tracking in metro areas: Malnets and others. In *Proceedings of the 17th ACM Conference on Computer and Communications Security*, CCS '10, pages 85–96, 2010.
- [59] The 802.11 subsystems for kernel developers. <https://www.kernel.org/doc/html/docs/80211/index.html>.
- [60] IEEE 802.11n-2009 data rates. https://en.wikipedia.org/wiki/IEEE_802.11n-2009#Data_rates.
- [61] IEEE. 802.11n-2009: Enhancements for higher throughput, 2009.
- [62] IEEE. 802.11ac-2013: Enhancements for very high throughput for operation in bands below 6 ghz, 2013.

- [63] SERGEY IOFFE AND CHRISTIAN SZEGEDY. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *Proceedings of the 32Nd International Conference on International Conference on Machine Learning - Volume 37*, ICML'15, pages 448–456. JMLR.org, 2015.
- [64] SHUJA JAMIL, SOHAIB KHAN, ANAS BASALAMAH, AND AHMED LBATH. Classifying smartphone screen on/off state based on wifi probe patterns. In *Proceedings of the 2016 ACM International Joint Conference on Pervasive and Ubiquitous Computing: Adjunct, UbiComp/ISWC'16 Adjunct*, pages 301–304, 2016.
- [65] KASTHURI JAYARAJAH, ZAMAN LANTRA, AND ARCHAN MISRA. Fusing wifi and video sensing for accurate group detection in indoor spaces. In *Proceedings of the 3rd International on Workshop on Physical Analytics, WPA '16*, pages 49–54. ACM, 2016.
- [66] WENJUN JIANG, CHENGLIN MIAO, FENGLONG MA, SHUOCHAO YAO, YAQING WANG, YE YUAN, HONGFEI XUE, CHEN SONG, XIN MA, DIMITRIOS KOUTSONIKOLAS, WENYAO XU, AND LU SU. Towards environment independent device free human activity recognition. In *Proceedings of the 24th Annual International Conference on Mobile Computing and Networking, MobiCom '18*, pages 289–304. ACM, 2018.
- [67] ZHIPING JIANG, JIZHONG ZHAO, XIANG-YANG LI, JINSONG HAN, AND WEI XI. Rejecting the attack: Source authentication for wi-fi management frames using csi information. In *2013 IEEE Conference on Computer Communications (INFOCOM)*, pages 2544–2552, April 2013.
- [68] KIRAN JOSHI, DINESH BHARADIA, MANIKANTA KOTARU, AND SACHIN KATTI. Video: Fine-grained device-free motion tracing using rf backscatter. In *Proceedings of the 12th USENIX Conference on Networked Systems Design and Implementation, NSDI'15*, pages 189–204, 2015.
- [69] K. KARAKAYALI, R. YATES, G. FOSCHINI, AND R. VALENZUELA. Optimum zero-forcing beamforming with per-antenna power constraints. In *IEEE ISIT*, 2007.
- [70] CHITRA R. KARANAM AND YASAMIN MOSTOFI. 3d through-wall imaging with unmanned aerial vehicles using wifi. In *Proceedings of the 16th ACM/IEEE International Conference on Information Processing in Sensor Networks, IPSN '17*, pages 131–142, 2017.

- [71] BRYCE KELLOGG, VAMSI TALLA, AND SHYAMNATH GOLLAKOTA. Bringing gesture recognition to all devices. In *Proceedings of the 11th USENIX Conference on Networked Systems Design and Implementation*, NSDI'14, pages 303–316, 2014.
- [72] MINKYONG KIM AND DAVID KOTZ. Modeling users' mobility among wifi access points. In *Papers Presented at the 2005 Workshop on Wireless Traffic Measurements and Modeling*, WiTMeMo '05, pages 19–24. USENIX Association, 2005.
- [73] SANG-CHUL KIM. Device-free activity recognition using csi big data analysis: A survey. In *2017 Ninth International Conference on Ubiquitous and Future Networks (ICUFN)*, pages 539–541, July 2017.
- [74] MIKKEL BAUN KJÆRGAARD AND PETTERI NURMI. Challenges for social sensing using wifi signals. In *Proceedings of the 1st ACM Workshop on Mobile Systems for Computational Social Science*, MCSS '12, pages 17–21, 2012.
- [75] MANIKANTA KOTARU, KIRAN JOSHI, DINESH BHARADIA, AND SACHIN KATTI. Spotfi: Decimeter level localization using wifi. In *Proceedings of the 2015 ACM Conference on Special Interest Group on Data Communication*, SIGCOMM '15, pages 269–282, 2015.
- [76] SWARUN KUMAR, DIEGO CIFUENTES, SHYAMNATH GOLLAKOTA, AND DINA KATABI. Bringing cross-layer mimo to today's wireless lans. In *Proceedings of the ACM SIGCOMM 2013 Conference on SIGCOMM*, SIGCOMM '13, pages 387–398, 2013.
- [77] H.L. LANE, R. HOFFMEISTER, AND B.J. BAHAN. *A Journey Into the Deaf-world*. Dawn-SignPress, 1996.
- [78] FAN LI, XIUXIU WANG, HUIJIE CHEN, KASHIF SHARIF, AND YU WANG. Clickleak: Keystroke leaks through multimodal sensors in cyber-physical social networks. *IEEE Access*, 5:27311–27321, 2017.
- [79] HONG LI, WEI YANG, JIANXIN WANG, YANG XU, AND LIUSHENG HUANG. Wifinger: Talk to your smart devices with finger-grained gesture. In *Proceedings of the 2016 ACM International Joint Conference on Pervasive and Ubiquitous Computing*, UbiComp '16, pages 250–261, 2016.

- [80] MENGYUAN LI, YAN MENG, JUNYI LIU, HAOJIN ZHU, XIAOHUI LIANG, YAO LIU, AND NA RUAN. When csi meets public wifi: Inferring your mobile phone password via wifi signals. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security, CCS '16*, pages 1068–1079, 2016.
- [81] SHENGJIE LI, XIANG LI, KAI NIU, HAO WANG, YUE ZHANG, AND DAQING ZHANG. Ar-alarm: An adaptive and robust intrusion detection system leveraging csi from commodity wi-fi. In *Enhanced Quality of Life and Smart Living*, pages 211–223. Springer International Publishing, 2017.
- [82] XIANG LI, DAQING ZHANG, QIN LV, JIE XIONG, SHENGJIE LI, YUE ZHANG, AND HONG MEI. Indotrack: Device-free indoor human tracking with commodity wi-fi. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.*, 1(3):72:1–72:22, September 2017.
- [83] JAIME LIEN, NICHOLAS GILLIAN, M. EMRE KARAGOZLER, PATRICK AMIHOOD, CARSTEN SCHWESIG, ERIK OLSON, HAKIM RAJA, AND IVAN POUPYREV. Soli: Ubiquitous gesture sensing with millimeter wave radar. *ACM Trans. Graph.*, 35(4):142:1–142:19, July 2016.
- [84] HANXIAO LIU, KAREN SIMONYAN, AND YIMING YANG. DARTS: differentiable architecture search. *arXiv:1806.09055*, 2018.
- [85] HONGBO LIU, YAN WANG, JIAN LIU, JIE YANG, AND YINGYING CHEN. Practical user authentication leveraging channel state information (csi). In *Proceedings of the 9th ACM Symposium on Information, Computer and Communications Security, ASIA CCS '14*, pages 389–400, 2014.
- [86] HONGBO LIU, YAN WANG, JIAN LIU, JIE YANG, YINGYING CHEN, AND H. VINCENT POOR. Authenticating users through fine-grained channel information. *IEEE Transactions on Mobile Computing*, 17(2):251–264, Feb 2018.
- [87] JIALIN LIU, LEI WANG, LINLIN GUO, JIAN FANG, BINGXIAN LU, AND WEI ZHOU. A research on csi-based human motion detection in complex scenarios. In *2017 IEEE 19th International Conference on e-Health Networking, Applications and Services (Healthcom)*, pages 1–6, Oct 2017.

- [88] JIAN LIU, YAN WANG, YINGYING CHEN, JIE YANG, XU CHEN, AND JERRY CHENG. Tracking vital signs during sleep leveraging off-the-shelf wifi. In *Proceedings of the 16th ACM International Symposium on Mobile Ad Hoc Networking and Computing, MobiHoc '15*, pages 267–276, 2015.
- [89] XUEFENG LIU, JIANNONG CAO, SHAOJIE TANG, AND JIAQI WEN. Wi-sleep: Contactless sleep monitoring via wifi signals. In *2014 IEEE Real-Time Systems Symposium*, pages 346–355, Dec 2014.
- [90] XUEFENG LIU, JIANNONG CAO, SHAOJIE TANG, JIAQI WEN, AND PENG GUO. Contactless respiration monitoring via off-the-shelf wifi devices. *IEEE Transactions on Mobile Computing*, 15(10):2466–2479, Oct 2016.
- [91] JIGUANG LV, DAPENG MAN, WU YANG, XIAOJIANG DU, AND MIAO YU. Robust wlan-based indoor intrusion detection using phy layer information. *IEEE Access*, 6(99):30117–30127, 2018.
- [92] HONGJIANG LYU, LINGHE KONG, CHENGZHANG LI, YUNXIN LIU, JIANSONG ZHANG, AND GUIHAI CHEN. Bikeloc: A real-time high-precision bicycle localization system using synthetic aperture radar. In *Proceedings of the First Asia-Pacific Workshop on Networking, APNet'17*, pages 57–63. ACM, 2017.
- [93] JUNYI MA, YUXIANG WANG, HAO WANG, YASHA WANG, AND DAQING ZHANG. When can we detect human respiration with commodity wifi devices? In *Proceedings of the 2016 ACM International Joint Conference on Pervasive and Ubiquitous Computing: Adjunct, UbiComp/ISWC'16 Adjunct*, pages 325–328, 2016.
- [94] YONGSEN MA, GANG ZHOU, AND SHUANGQUAN WANG. Wifi sensing with channel state information: A survey. *ACM Comput. Surv.*, 52(3):46:1–46:36, June 2019.
- [95] YONGSEN MA, GANG ZHOU, SHUANGQUAN WANG, HONGYANG ZHAO, AND WOOSUB JUNG. Signfi: Sign language recognition using wifi. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.*, 2(1):23:1–23:21, March 2018.

- [96] SAURABH MAHESHWARI AND ANIL K. TIWARI. Walking parameters estimation through channel state information preliminary results. In *2015 9th International Conference on Signal Processing and Communication Systems (ICSPCS)*, pages 1–8, Dec 2015.
- [97] RAJESH B. MAPARI AND GOVIND KHARAT. American static signs recognition using leap motion sensor. In *Proceedings of the Second International Conference on Information and Communication Technology for Competitive Strategies, ICTCS '16*, pages 67:1–67:5, 2016.
- [98] ANDREAS MARCALETTI, MAURIZIO REA, DOMENICO GIUSTINIANO, VINCENT LENDERS, AND AYMEN FAKHREDDINE. Filtering noisy 802.11 time-of-flight ranging measurements. In *ACM CoNEXT*, 2014.
- [99] ALEX T. MARIAKAKIS, SOUVIK SEN, JEONGKEUN LEE, AND KYU-HAN KIM. Sail: Single access point-based indoor localization. In *ACM MobiSys*, 2014.
- [100] PEDRO MELGAREJO, XINYU ZHANG, PARAMESWARAN RAMANATHAN, AND DAVID CHU. Leveraging directional antenna capabilities for fine-grained gesture recognition. In *Proceedings of the 2014 ACM International Joint Conference on Pervasive and Ubiquitous Computing, UbiComp '14*, pages 541–551, 2014.
- [101] Minstrel rate control algorithm. <https://goo.gl/T3AFV2>.
- [102] HESSAM MOHAMMADMORADI, SHENGRONG YIN, AND OMPRAKASH GNAWALI. Room occupancy estimation through wifi, uwb, and light sensors mounted on doorways. In *Proceedings of the 2017 International Conference on Smart Digital Environment, ICSDE '17*, pages 27–34. ACM, 2017.
- [103] P. MOLCHANOV, S. GUPTA, K. KIM, AND K. PULLI. Multi-sensor system for driver’s hand-gesture recognition. In *2015 11th IEEE International Conference and Workshops on Automatic Face and Gesture Recognition (FG)*, volume 1, pages 1–8, May 2015.
- [104] MOTIONSAVVY. Motionsavvy uni, 2017. Retrieved July 23, 2017 from <http://www.motionsavvy.com/uni.html>.
- [105] D. NAGLOT AND M. KULKARNI. Real time sign language recognition using the leap motion controller. In *2016 International Conference on Inventive Computation Technologies (ICICT)*, volume 3, pages 1–5, August 2016.

- [106] RAJALAKSHMI NANDAKUMAR, VIKRAM IYER, DESNEY TAN, AND SHYAMNATH GOLAKOTA. Fingerio: Using active sonar for fine-grained finger tracking. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*, CHI '16, pages 1515–1525, 2016.
- [107] UNIVERSITY OF WASHINGTON. Signaloud demo, 2016. Retrieved July 23, 2017 from <https://youtu.be/4uY-MyoRq4c>.
- [108] KAZUYA OHARA, TAKUYA MAEKAWA, AND YASUYUKI MATSUSHITA. Detecting state changes of indoor everyday objects using wi-fi channel state information. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.*, 1(3):88:1–88:28, September 2017.
- [109] SAMEERA PALIPANA, PIYUSH AGRAWAL, AND DIRK PESCH. Channel state information based human presence detection using non-linear techniques. In *Proceedings of the 3rd ACM International Conference on Systems for Energy-Efficient Built Environments*, BuildSys '16, pages 177–186, 2016.
- [110] SAMEERA PALIPANA, DAVID ROJAS, PIYUSH AGRAWAL, AND DIRK PESCH. Falldefi: Ubiquitous fall detection using commodity wi-fi devices. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.*, 1(4):155:1–155:25, January 2018.
- [111] TAIWOO PARK, JINWON LEE, INSEOK HWANG, CHUNGKUK YOO, LAMA NACHMAN, AND JUNEHWA SONG. E-gesture: A collaborative architecture for energy-efficient gesture recognition with hand-worn sensor and mobile devices. In *Proceedings of the 9th ACM Conference on Embedded Networked Sensor Systems*, SenSys '11, pages 260–273, 2011.
- [112] ANINDYA S. PAUL, ERIC A. WAN, FATEMA ADENWALA, ERICH SCHAFFERMEYER, NICK PREISER, JEFFREY KAYE, AND PETER G. JACOBS. Mobilerf: A robust device-free tracking system based on a hybrid neural network hmm classifier. In *Proceedings of the 2014 ACM International Joint Conference on Pervasive and Ubiquitous Computing*, UbiComp '14, pages 159–170, 2014.
- [113] HONGJIAN PENG AND WEIJIA JIA. Wifind: Driver fatigue detection with fine-grained wi-fi signal features. In *GLOBECOM 2017 - 2017 IEEE Global Communications Conference*, pages 1–6, Dec 2017.

- [114] ELDAD PERAHIA AND ROBERT STACEY. *Next Generation Wireless LANs: 802.11n and 802.11ac*. Cambridge University Press, 2nd edition, 2013.
- [115] HIEU PHAM, MELODY Y. GUAN, BARRET ZOPH, QUOC V. LE, AND JEFF DEAN. Efficient neural architecture search via parameter sharing. *arXiv:1802.03268*, 2018.
- [116] LIONEL PIGOU, SANDER DIELEMAN, PIETER-JAN KINDERMANS, AND BENJAMIN SCHRAUWEN. *Sign Language Recognition Using Convolutional Neural Networks*, pages 572–578. Springer International Publishing, 2015.
- [117] PICHAYA PRASERTSUNG AND TEERAYUT HORANONT. How does coffee shop get crowded?: Using wifi footprints to deliver insights into the success of promotion. In *Proceedings of the 2017 ACM International Symposium on Wearable Computers, UbiComp/ISWC’17 Adjunct*, pages 421–426, 2017.
- [118] QIFAN PU, SIDHANT GUPTA, SHYAMNATH GOLLAKOTA, AND SHWETAK PATEL. Whole-home gesture recognition using wireless signals. In *Proceedings of the 19th Annual International Conference on Mobile Computing and Networking, MobiCom ’13*, pages 27–38, 2013.
- [119] KUN QIAN, CHENSHU WU, ZHENG YANG, YUNHAO LIU, FUGUI HE, AND TIANZHANG XING. Enabling contactless detection of moving humans with dynamic speeds using csi. *ACM Trans. Embed. Comput. Syst.*, 17(2):52:1–52:18, January 2018.
- [120] KUN QIAN, CHENSHU WU, ZHENG YANG, YUNHAO LIU, AND KYLE JAMIESON. Widar: Decimeter-level passive tracking via velocity monitoring with commodity wi-fi. In *Proceedings of the 18th ACM International Symposium on Mobile Ad Hoc Networking and Computing, Mobihoc ’17*, pages 6:1–6:10, 2017.
- [121] KUN QIAN, CHENSHU WU, ZHENG YANG, YUNHAO LIU, AND ZIMU ZHOU. Pads: Passive detection of moving targets with dynamic speed using phy layer information. In *2014 20th IEEE International Conference on Parallel and Distributed Systems (ICPADS)*, pages 1–8, Dec 2014.
- [122] KUN QIAN, CHENSHU WU, ZHENG YANG, CHAOFAN YANG, AND YUNHAO LIU. Decimeter level passive tracking with wifi. In *Proceedings of the 3rd Workshop on Hot Topics in Wireless, HotWireless ’16*, pages 44–48. ACM, 2016.

- [123] KUN QIAN, CHENSHU WU, ZIMU ZHOU, YUE ZHENG, ZHENG YANG, AND YUNHAO LIU. Inferring motion direction using commodity wi-fi for interactive exergames. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems*, CHI '17, pages 1961–1972. ACM, 2017.
- [124] LUIS QUESADA, GUSTAVO LÓPEZ, AND LUIS A. GUERRERO. *Sign Language Recognition Using Leap Motion*, pages 277–288. Springer International Publishing, 2015.
- [125] MUNEEBA RAJA, VIVIANE GHADERI, AND STEPHAN SIGG. Wibot! in-vehicle behaviour and gesture recognition using wireless network edge. In *2018 IEEE 38th International Conference on Distributed Computing Systems (ICDCS)*, pages 376–387, July 2018.
- [126] XIONGBIN RAO AND VINCENT K. N. LAU. Distributed compressive CSIT estimation and feedback for FDD multi-user massive MIMO systems. *IEEE Transactions on Signal Processing*, 62(12):3261–3271, 2014.
- [127] LENIN RAVINDRANATH, CALVIN NEWPORT, HARI BALAKRISHNAN, AND SAMUEL MADDEN. Improving wireless network performance using sensor hints. In *USENIX NSDI*, 2011.
- [128] CHARLES REIS, RATUL MAHAJAN, MAYA RODRIG, DAVID WETHERALL, AND JOHN ZAHORJAN. Measurement-based models of delivery and interference in static wireless networks. In *ACM SIGCOMM*, 2006.
- [129] ETTUS RESEARCH. Usrp software defined radio device, 2004. <https://www.ettus.com>.
- [130] WENJIE RUAN, QUAN Z. SHENG, LEI YANG, TAO GU, PEIPEI XU, AND LONGFEI SHANG-GUAN. Audiogest: Enabling fine-grained hand gesture detection by decoding echo signal. In *Proceedings of the 2016 ACM International Joint Conference on Pervasive and Ubiquitous Computing*, UbiComp '16, pages 474–485, 2016.
- [131] S. K. SAHA, P. DESHPANDE, P. P. INAMDAR, R. K. SHESHADRI, AND D. KOUTSONIKOLAS. Power-throughput tradeoffs of 802.11n/ac in smartphones. In *IEEE INFOCOM*, 2015.
- [132] C. SAVUR AND F. SAHIN. Real-time american sign language recognition system using surface emg signal. In *2015 IEEE 14th International Conference on Machine Learning and Applications (ICMLA)*, pages 497–502, Dec 2015.

- [133] SCIKIT LEARN. Classification metrics, 2019. https://scikit-learn.org/stable/modules/model_evaluation.html.
- [134] SOUVIK SEN, JEONGKEUN LEE, KYU-HAN KIM, AND PAUL CONGDON. Avoiding multipath to revive inbuilding WiFi localization. In *ACM MobiSys*, 2013.
- [135] JIACHENG SHANG AND JIE WU. Fine-grained vital signs estimation using commercial wi-fi devices. In *Proceedings of the Eighth Wireless of the Students, by the Students, and for the Students Workshop*, S3, pages 30–32. ACM, 2016.
- [136] JIACHENG SHANG AND JIE WU. A robust sign language recognition system with multiple wi-fi devices. In *Proceedings of the Workshop on Mobility in the Evolving Internet Architecture*, MobiArch '17, pages 19–24, 2017.
- [137] CONG SHI, JIAN LIU, HONGBO LIU, AND YINGYING CHEN. Smart user authentication through actuation of daily activities leveraging wifi-enabled iot. In *Proceedings of the 18th ACM International Symposium on Mobile Ad Hoc Networking and Computing*, Mobihoc '17, pages 5:1–5:10, 2017.
- [138] SIGNALL. Signall prototype, 2017. Retrieved July 23, 2017 from <http://www.signall.us>.
- [139] SOLIDRUN. Hummingboard, 2019. <https://www.solid-run.com/nxp-family/hummingboard/>.
- [140] ELAHE SOLTANAGHAEI, AVINASH KALYANARAMAN, AND KAMIN WHITEHOUSE. Peripheral wifi vision: Exploiting multipath reflections for more sensitive human sensing. In *Proceedings of the 4th International on Workshop on Physical Analytics*, WPA '17, pages 13–18. ACM, 2017.
- [141] STARTASL. Basic words in sign language, 2017. Retrieved July 14, 2017 from <https://www.startasl.com/basic-words-in-sign-language.html>.
- [142] CHAO SUN, TIANZHU ZHANG, AND CHANGSHENG XU. Latent support vector machine modeling for sign language recognition with kinect. *ACM Trans. Intell. Syst. Technol.*, 6(2):20:1–20:20, March 2015.

- [143] LI SUN, SOUVIK SEN, AND DIMITRIOS KOUTSONIKOLAS. Bringing mobility-awareness to WLANs using PHY layer information. In *ACM CoNEXT*, 2014.
- [144] LI SUN, SOUVIK SEN, DIMITRIOS KOUTSONIKOLAS, AND KYU-HAN KIM. 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*, MobiCom '15, pages 77–89, 2015.
- [145] RICHARD S SUTTON AND ANDREW G BARTO. *Reinforcement learning: An introduction*. MIT press, 2ed edition, 2018.
- [146] SHENG TAN AND JIE YANG. 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*, MobiHoc '16, pages 201–210, 2016.
- [147] YONGLONG TIAN, GUANG-HE LEE, HAO HE, CHEN-YU HSU, AND DINA KATABI. Rf-based fall monitoring using convolutional neural networks. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.*, 2(3):137:1–137:24, September 2018.
- [148] DAVID TSE AND PRAMOD VISWANATH. *Fundamentals of Wireless Communication*. Cambridge University Press, 2005.
- [149] RICE UNIVERSITY. Warp: Wireless open access research platform, 2006. <https://warpproject.org/trac>.
- [150] DEEPAK VASISHT, SWARUN KUMAR, AND DINA KATABI. Decimeter-level localization with a single wifi access point. In *Proceedings of the 13th USENIX Conference on Networked Systems Design and Implementation*, NSDI'16, pages 165–178, 2016.
- [151] DEEPAK VASISHT, SWARUN KUMAR, AND DINA KATABI. Decimeter-level localization with a single WiFi access point. In *USENIX NSDI*, 2016.
- [152] DEEPAK VASISHT, SWARUN KUMAR, HARIHARAN RAHUL, AND DINA KATABI. Eliminating channel feedback in next-generation cellular networks. In *Proceedings of the 2016 Conference on ACM SIGCOMM 2016 Conference*, SIGCOMM '16, pages 398–411, New York, NY, USA, 2016. ACM.

- [153] RAGHAV H. VENKATNARAYAN, GRIFFIN PAGE, AND MUHAMMAD SHAHZAD. Multi-user gesture recognition using wifi. In *Proceedings of the 16th Annual International Conference on Mobile Systems, Applications, and Services*, MobiSys '18, pages 401–413, 2018.
- [154] ADITYA VIRMANI AND MUHAMMAD SHAHZAD. Position and orientation agnostic gesture recognition using wifi. In *Proceedings of the 15th Annual International Conference on Mobile Systems, Applications, and Services*, MobiSys '17, pages 252–264, 2017.
- [155] GUANHUA WANG, YONGPAN ZOU, ZIMU ZHOU, KAISHUN WU, AND LIONEL M. NI. We can hear you with wi-fi! In *Proceedings of the 20th Annual International Conference on Mobile Computing and Networking*, MobiCom '14, pages 593–604, 2014.
- [156] HAO WANG, DAQING ZHANG, JUNYI MA, YASHA WANG, YUXIANG WANG, DAN WU, TAO GU, AND BING XIE. Human respiration detection with commodity wifi devices: Do user location and body orientation matter? In *Proceedings of the 2016 ACM International Joint Conference on Pervasive and Ubiquitous Computing*, UbiComp '16, pages 25–36, 2016.
- [157] HAO WANG, DAQING ZHANG, YASHA WANG, JUNYI MA, YUXIANG WANG, AND SHENGJIE LI. Rt-fall: A real-time and contactless fall detection system with commodity wifi devices. *IEEE Transactions on Mobile Computing*, 16(2):511–526, February 2017.
- [158] JIE WANG, LIMING ZHANG, QINGHUA GAO, MIAO PAN, AND HONGYU WANG. Device-free wireless sensing in complex scenarios using spatial structural information. *IEEE Transactions on Wireless Communications*, 17(4):2432–2442, April 2018.
- [159] JU WANG, HONGBO JIANG, JIE XIONG, KYLE JAMIESON, XIAOJIANG CHEN, DINGYI FANG, AND BINBIN XIE. Lifis: Low human-effort, device-free localization with fine-grained subcarrier information. In *Proceedings of the 22Nd Annual International Conference on Mobile Computing and Networking*, MobiCom '16, pages 243–256. ACM, 2016.
- [160] JUE WANG, DEEPAK VASISHT, AND DINA KATABI. Rf-idraw: Virtual touch screen in the air using rf signals. In *Proceedings of the 2014 ACM Conference on SIGCOMM*, SIGCOMM '14, pages 235–246, 2014.

- [161] PEI WANG, BIN GUO, TONG XIN, ZHU WANG, AND ZHIWEN YU. Tinsense: Multi-user respiration detection using wi-fi csi signals. In *2017 IEEE 19th International Conference on e-Health Networking, Applications and Services (Healthcom)*, pages 1–6, Oct 2017.
- [162] QI WANG, HAO FENG, L.J. CIMINI, L.J. GREENSTEIN, D.S. CHAN, AND A. HEDAYAT. Comparison of quantization techniques for downlink multi-user MIMO channels with limited feedback. *IEEE Wireless Communications Letters*, 3(2):165–168, 2014.
- [163] WEI WANG, YINGJIE CHEN, AND QIAN ZHANG. Privacy-preserving location authentication in wi-fi networks using fine-grained physical layer signatures. *IEEE Transactions on Wireless Communications*, 15(2):1218–1225, Feb 2016.
- [164] WEI WANG, ALEX X. LIU, AND MUHAMMAD SHAHZAD. Gait recognition using wifi signals. In *Proceedings of the 2016 ACM International Joint Conference on Pervasive and Ubiquitous Computing, UbiComp '16*, pages 363–373, 2016.
- [165] WEI WANG, ALEX X. LIU, MUHAMMAD SHAHZAD, KANG LING, AND SANGLU LU. Understanding and modeling of wifi signal based human activity recognition. In *Proceedings of the 21st Annual International Conference on Mobile Computing and Networking, MobiCom '15*, pages 65–76. ACM, 2015.
- [166] WEI WANG, ALEX X. LIU, MUHAMMAD SHAHZAD, KANG LING, AND SANGLU LU. Device-free human activity recognition using commercial wifi devices. *IEEE Journal on Selected Areas in Communications*, 35(5):1118–1131, May 2017.
- [167] WEI WANG, ALEX X. LIU, AND KE SUN. Device-free gesture tracking using acoustic signals. In *Proceedings of the 22Nd Annual International Conference on Mobile Computing and Networking, MobiCom '16*, pages 82–94, 2016.
- [168] XUYU WANG, CHAO YANG, AND SHIWEN MAO. Phasebeat: Exploiting csi phase data for vital sign monitoring with commodity wifi devices. In *2017 IEEE 37th International Conference on Distributed Computing Systems (ICDCS)*, pages 1230–1239, June 2017.
- [169] XUYU WANG, CHAO YANG, AND SHIWEN MAO. Tensorbeat: Tensor decomposition for monitoring multi-person breathing beats with commodity wifi. *arXiv:1702.02046*, 2017.

- [170] YAN WANG, JIAN LIU, YINGYING CHEN, MARCO GRUTESER, JIE YANG, AND HONGBO LIU. E-eyes: Device-free location-oriented activity identification using fine-grained wifi signatures. In *Proceedings of the 20th Annual International Conference on Mobile Computing and Networking*, MobiCom '14, pages 617–628. ACM, 2014.
- [171] YAN WANG, JIE YANG, YINGYING CHEN, HONGBO LIU, MARCO GRUTESER, AND RICHARD P. MARTIN. Tracking human queues using single-point signal monitoring. In *Proceedings of the 12th Annual International Conference on Mobile Systems, Applications, and Services*, MobiSys '14, pages 42–54. ACM, 2014.
- [172] YAN WANG, JIE YANG, HONGBO LIU, YINGYING CHEN, MARCO GRUTESER, AND RICHARD P. MARTIN. Measuring human queues using wifi signals. In *Proceedings of the 19th Annual International Conference on Mobile Computing & Networking*, MobiCom '13, pages 235–238. ACM, 2013.
- [173] YI WANG, XINLI JIANG, RONGYU CAO, AND XIYANG WANG. Robust indoor human activity recognition using wireless signals. *Sensors*, 15(7):17195–17208, 2015.
- [174] ZHU WANG, BIN GUO, ZHIWEN YU, AND XINGSHE ZHOU. Wi-fi csi based behavior recognition: From signals, actions to activities. *arXiv:1712.00146*, 2017.
- [175] HIKARU WATANABE, MASAHIRO MOCHIZUKI, KAZUYA MURAO, AND NOBUHIKO NISHIO. A recognition method for continuous gestures with an accelerometer. In *Proceedings of the 2016 ACM International Joint Conference on Pervasive and Ubiquitous Computing: Adjunct*, UbiComp '16, pages 813–822, 2016.
- [176] BO WEI, WEN HU, MINGRUI YANG, AND CHUN TUNG CHOU. Radio-based device-free activity recognition with radio frequency interference. In *Proceedings of the 14th International Conference on Information Processing in Sensor Networks*, IPSN '15, pages 154–165, 2015.
- [177] TENG WEI, SHU WANG, ANFU ZHOU, AND XINYU ZHANG. Acoustic eavesdropping through wireless vibrometry. In *Proceedings of the 21st Annual International Conference on Mobile Computing and Networking*, MobiCom '15, pages 130–141. ACM, 2015.

- [178] TENG WEI AND XINYU ZHANG. mtrack: High-precision passive tracking using millimeter wave radios. In *Proceedings of the 21st Annual International Conference on Mobile Computing and Networking*, MobiCom '15, pages 117–129, 2015.
- [179] HONGYI WEN, JULIAN RAMOS ROJAS, AND ANIND K. DEY. Serendipity: Finger gesture recognition using an off-the-shelf smartwatch. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*, CHI '16, pages 3847–3851, 2016.
- [180] ERIC WENGROWSKI. A survey on device-free passive localization and gesture recognition via body wave reflections. Technical report, Rutgers University, 2014. <https://pdfs.semanticscholar.org/24c6/5db8fd18a29037147ccabca09e2196ea87e5.pdf>.
- [181] What does the OS X Activity Monitor's "Energy Impact" actually measure? <https://google.com/search?q=OS+X+Activity+Monitor+Energy+Impact>.
- [182] L. P. WITHERS, R. M. TAYLOR, AND D. M. WARME. Echo-MIMO: A two-way channel training method for matched cooperative beamforming. *IEEE Transactions on Signal Processing*, 56(9):4419–4432, 2008.
- [183] MYOUNGGYU WON, SHAOHU ZHANG, AND SANG H. SON. Witraffic: Low-cost and non-intrusive traffic monitoring system using wifi. In *2017 26th International Conference on Computer Communication and Networks (ICCCN)*, pages 1–9, July 2017.
- [184] CHENSHU WU, ZHENG YANG, ZIMU ZHOU, XUEFENG LIU, YUNHAO LIU, AND JIANNONG CAO. Non-invasive detection of moving and stationary human with wifi. *IEEE Journal on Selected Areas in Communications*, 33(11):2329–2342, Nov 2015.
- [185] CHENSHU WU, ZHENG YANG, ZIMU ZHOU, KUN QIAN, YUNHAO LIU, AND MINGYAN LIU. Phaseu: Real-time los identification with wifi. In *2015 IEEE Conference on Computer Communications (INFOCOM)*, pages 2038–2046, April 2015.
- [186] DAN WU, DAQING ZHANG, CHENREN XU, HAO WANG, AND XIANG LI. Device-free wifi human sensing: From pattern-based to model-based approaches. *IEEE Communications Magazine*, 55(10):91–97, Oct 2017.
- [187] DAN WU, DAQING ZHANG, CHENREN XU, YASHA WANG, AND HAO WANG. Widir: Walking direction estimation using wireless signals. In *Proceedings of the 2016 ACM International*

- Joint Conference on Pervasive and Ubiquitous Computing*, UbiComp '16, pages 351–362, 2016.
- [188] KAISHUN WU. Wi-metal: Detecting metal by using wireless networks. In *2016 IEEE International Conference on Communications (ICC)*, pages 1–6, May 2016.
- [189] WEI XI, DONG HUANG, KUN ZHAO, YUBO YAN, YUANHANG CAI, RONG MA, AND DENG CHEN. Device-free human activity recognition using csi. In *Proceedings of the 1st Workshop on Context Sensing and Activity Recognition*, CSAR '15, pages 31–36. ACM, 2015.
- [190] WEI XI, CHEN QIAN, JINSONG HAN, KUN ZHAO, SHENG ZHONG, XIANG-YANG LI, AND JIZHONG ZHAO. Instant and robust authentication and key agreement among mobile devices. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, CCS '16, pages 616–627, 2016.
- [191] WEI XI, JIZHONG ZHAO, XIANG-YANG LI, KUN ZHAO, SHAOJIE TANG, XUE LIU, AND ZHIPING JIANG. Electronic frog eye: Counting crowd using wifi. In *2014 IEEE Conference on Computer Communications (INFOCOM)*, pages 361–369, April 2014.
- [192] FU XIAO, JING CHEN, XIAO HUI XIE, LINQING GUI, JUAN LI SUN, AND WANG RUCHUAN. Seare: A system for exercise activity recognition and quality evaluation based on green sensing. *IEEE Transactions on Emerging Topics in Computing*, 2018.
- [193] FU XIAO, XIAOHUI XIE, HAI ZHU, LIJUAN SUN, AND RUCHUAN WANG. Invisible cloak fails: Csi-based passive human detection. In *Proceedings of the 1st Workshop on Context Sensing and Activity Recognition*, CSAR '15, pages 19–23. ACM, 2015.
- [194] JIANG XIAO, ZIMU ZHOU, YOUWEN YI, AND LIONEL M. NI. A survey on wireless indoor localization from the device perspective. *ACM Comput. Surv.*, 49(2):25:1–25:31, June 2016.
- [195] XIUFENG XIE, XINYU ZHANG, AND KARTHIKEYAN SUNDARESAN. Adaptive feedback compression for MIMO networks. In *ACM MobiCom*, 2013.
- [196] YAXIONG XIE, ZHENJIANG LI, AND MO LI. Precise power delay profiling with commodity wifi. In *Proceedings of the 21st Annual International Conference on Mobile Computing and Networking*, MobiCom '15, pages 53–64. ACM, 2015.

- [197] YAXIONG XIE, ZHENJIANG LI, AND MO LI. Precise power delay profiling with commodity WiFi. In *ACM MobiCom*, 2015.
- [198] TONG XIN, BIN GUO, ZHU WANG, MINGYANG LI, AND ZHIWEN YU. Freesense: Indoor human identification with wifi signals. *arXiv:1608.03430*, 2016.
- [199] JIE XIONG AND KYLE JAMIESON. Securearray: Improving wifi security with fine-grained physical-layer information. In *Proceedings of the 19th Annual International Conference on Mobile Computing & Networking*, MobiCom '13, pages 441–452. ACM, 2013.
- [200] JIE XIONG, KARTHIKEYAN SUNDARESAN, AND KYLE JAMIESON. ToneTrack: Leveraging frequency-agile radios for time-based indoor wireless localization. In *ACM MobiCom*, 2015.
- [201] CHAO XU, PARTH H. PATHAK, AND PRASANT MOHAPATRA. Finger-writing with smartwatch: A case for finger and hand gesture recognition using smartwatch. In *Proceedings of the 16th International Workshop on Mobile Computing Systems and Applications*, HotMobile '15, pages 9–14, 2015.
- [202] YANG XU, WEI YANG, JIANXIN WANG, XING ZHOU, HONG LI, AND LIUSHENG HUANG. Wistep: Device-free step counting with wifi signals. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.*, 1(4):172:1–172:23, January 2018.
- [203] HAO YANG, LICAI ZHU, AND WEIPENG LV. A hci motion recognition system based on channel state information with fine granularity. In *Wireless Algorithms, Systems, and Applications*, pages 780–790. Springer International Publishing, 2017.
- [204] ZHENG YANG, ZIMU ZHOU, AND YUNHAO LIU. From rssi to csi: Indoor localization via channel response. *ACM Comput. Surv.*, 46(2):25:1–25:32, December 2013.
- [205] SIAMAK YOUSEFI, HIROKAZU NARUI, SANKALP DAYAL, STEFANO ERMON, AND SHAHROKH VALAEE. A survey on behavior recognition using wifi channel state information. *IEEE Communications Magazine*, 55(10):98–104, Oct 2017.
- [206] HANG YU, LIN ZHONG, ASHUTOSH SABHARWAL, AND DAVID KAO. Beamforming on mobile devices: A first study. In *ACM MobiCom*, 2011.

- [207] NAN YU, WEI WANG, ALEX X. LIU, AND LINGTAO KONG. Qgesture: Quantifying gesture distance and direction with wifi signals. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.*, 2(1):51:1–51:23, March 2018.
- [208] SANGKI YUN, YI-CHAO CHEN, AND LILI QIU. Turning a mobile device into a mouse in the air. In *Proceedings of the 13th Annual International Conference on Mobile Systems, Applications, and Services*, MobiSys '15, pages 15–29. ACM, 2015.
- [209] SANGKI YUN, YI-CHAO CHEN, HUIHUANG ZHENG, LILI QIU, AND WENGUANG MAO. Strata: Fine-grained acoustic-based device-free tracking. In *Proceedings of the 15th Annual International Conference on Mobile Systems, Applications, and Services*, MobiSys '17, pages 15–28, 2017.
- [210] ZAHOOR ZAFRULLA, HELENE BRASHEAR, THAD STARNER, HARLEY HAMILTON, AND PETER PRESTI. American sign language recognition with the kinect. In *Proceedings of the 13th International Conference on Multimodal Interfaces*, ICMI '11, pages 279–286, 2011.
- [211] YUNZE ZENG, PARTH H. PATHAK, AND PRASANT MOHAPATRA. Analyzing shopper's behavior through wifi signals. In *Proceedings of the 2Nd Workshop on Workshop on Physical Analytics*, WPA '15, pages 13–18. ACM, 2015.
- [212] YUNZE ZENG, PARTH H. PATHAK, AND PRASANT MOHAPATRA. Wiwho: Wifi-based person identification in smart spaces. In *Proceedings of the 15th International Conference on Information Processing in Sensor Networks*, IPSN '16, pages 4:1–4:12. IEEE Press, 2016.
- [213] YUNZE ZENG, PARTH H. PATHAK, CHAO XU, AND PRASANT MOHAPATRA. Your ap knows how you move: Fine-grained device motion recognition through wifi. In *Proceedings of the 1st ACM Workshop on Hot Topics in Wireless*, HotWireless '14, pages 49–54, 2014.
- [214] DAQING ZHANG, HAO WANG, YASHA WANG, AND JUNYI MA. Anti-fall: A non-intrusive and real-time fall detector leveraging csi from commodity wifi devices. In *Inclusive Smart Cities and e-Health*, pages 181–193. Springer International Publishing, 2015.
- [215] DAQING ZHANG, HAO WANG, AND DAN WU. Toward centimeter-scale human activity sensing with wi-fi signals. *Computer*, 50(1):48–57, Jan 2017.

- [216] FENG ZHANG, CHEN CHEN, BEIBEI WANG, AND K. J. RAY LIU. Wispeed: A statistical electromagnetic approach for device-free indoor speed estimation. *IEEE Internet of Things Journal*, 2018.
- [217] FUSANG ZHANG, DAQING ZHANG, JIE XIONG, HAO WANG, KAI NIU, BEIHONG JIN, AND YUXIANG WANG. From fresnel diffraction model to fine-grained human respiration sensing with commodity wi-fi devices. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.*, 2(1):53:1–53:23, March 2018.
- [218] JIN ZHANG, BO WEI, WEN HU, SALII S. KANHERE, AND ARIEL TAN. Human identification using wifi signal. In *2016 IEEE International Conference on Pervasive Computing and Communication Workshops (PerCom Workshops)*, pages 1–2, March 2016.
- [219] OUYANG ZHANG AND KANNAN SRINIVASAN. Mudra: User-friendly fine-grained gesture recognition using wifi signals. In *Proceedings of the 12th International on Conference on Emerging Networking EXperiments and Technologies*, CoNEXT '16, pages 83–96, 2016.
- [220] XIANG ZHANG, RUKHSANA RUBY, JINFENG LONG, LU WANG, ZHONG MING, AND KAISHUN WU. Wihumidity: A novel csi-based humidity measurement system. In *Smart Computing and Communication*, pages 537–547. Springer International Publishing, 2017.
- [221] CHEN ZHAO, KE-YU CHEN, MD TANVIR ISLAM AUMI, SHWETAK PATEL, AND MATTHEW S. REYNOLDS. Sideswipe: Detecting in-air gestures around mobile devices using actual gsm signal. In *Proceedings of the 27th Annual ACM Symposium on User Interface Software and Technology*, UIST '14, pages 527–534, 2014.
- [222] XIAOLONG ZHENG, JILIANG WANG, LONGFEI SHANGGUAN, ZIMU ZHOU, AND YUNHAO LIU. Smokey: Ubiquitous smoking detection with commercial wifi infrastructures. In *2016 IEEE Conference on Computer Communications (INFOCOM)*, pages 1–9, April 2016.
- [223] XIAOLONG ZHENG, JILIANG WANG, LONGFEI SHANGGUAN, ZIMU ZHOU, AND YUNHAO LIU. Design and implementation of a csi-based ubiquitous smoking detection system. *IEEE/ACM Transactions on Networking*, 25(6):3781–3793, Dec 2017.
- [224] ANFU ZHOU, TENG WEI, XINYU ZHANG, MIN LIU, AND ZHONGCHENG LI. Signpost: Scalable mu-mimo signaling with zero csi feedback. In *Proceedings of the 16th ACM In-*

- ternational Symposium on Mobile Ad Hoc Networking and Computing*, MobiHoc '15, pages 327–336, New York, NY, USA, 2015. ACM.
- [225] MENGYU ZHOU, DAN PEI, KAIXIN SUI, AND THOMAS MOSCIBRODA. Mining crowd mobility and wifi hotspots on a densely-populated campus. In *Proceedings of the 2017 ACM International Symposium on Wearable Computers*, UbiComp/ISWC'17 Adjunct, pages 427–431, 2017.
- [226] MENGYU ZHOU, KAIXIN SUI, MINGHUA MA, YOUJIAN ZHAO, DAN PEI, AND THOMAS MOSCIBRODA. Mobicamp: A campus-wide testbed for studying mobile physical activities. In *Proceedings of the 3rd International on Workshop on Physical Analytics*, WPA '16, pages 1–6. ACM, 2016.
- [227] QIZHEN ZHOU, CHENSHU WU, JIANCHUN XING, JUELONG LI, ZHENG YANG, AND QIL-
IANG YANG. Wi-dog: Monitoring school violence with commodity wifi devices. In *Wireless Algorithms, Systems, and Applications*, pages 47–59. Springer International Publishing, 2017.
- [228] QIZHEN ZHOU, JIANCHUN XING, JUELONG LI, AND QILIAN YANG. A device-free number gesture recognition approach based on deep learning. In *2016 12th International Conference on Computational Intelligence and Security (CIS)*, pages 57–63, Dec 2016.
- [229] RUI ZHOU, XIANG LU, PENG BIAO ZHAO, AND JIE SONG CHEN. Device-free presence detection and localization with svm and csi fingerprinting. *IEEE Sensors Journal*, 17(23):7990–7999, Dec 2017.
- [230] ZIMU ZHOU, ZHENG YANG, CHENSHU WU, LONGFEI SHANGGUAN, AND YUNHAO LIU. Omnidirectional coverage for device-free passive human detection. *IEEE Transactions on Parallel and Distributed Systems*, 25(7):1819–1829, July 2014.
- [231] ZIMU ZHOU, ZHENG YANG, CHENSHU WU, WEI SUN, AND YUNHAO LIU. Lifi: Line-of-sight identification with wifi. In *2014 IEEE Conference on Computer Communications (INFOCOM)*, pages 2688–2696, April 2014.
- [232] DALI ZHU, NA PANG, GANG LI, AND SHAOWU LIU. Notifi: A ubiquitous wifi-based abnormal activity detection system. In *2017 International Joint Conference on Neural Networks (IJCNN)*, pages 1766–1773, May 2017.

- [233] HAI ZHU, FU XIAO, LIJUAN SUN, RUCHUAN WANG, AND PANLONG YANG. R-ttwd: Robust device-free through-the-wall detection of moving human with wifi. *IEEE Journal on Selected Areas in Communications*, 35(5):1090–1103, May 2017.
- [234] YANZI ZHU, YUANSUN YAO, BEN Y. ZHAO, AND HAITAO ZHENG. Object recognition and navigation using a single networking device. In *Proceedings of the 15th Annual International Conference on Mobile Systems, Applications, and Services, MobiSys '17*, pages 265–277. ACM, 2017.
- [235] YANZI ZHU, YIBO ZHU, BEN Y. ZHAO, AND HAITAO ZHENG. Reusing 60ghz radios for mobile radar imaging. In *Proceedings of the 21st Annual International Conference on Mobile Computing and Networking, MobiCom '15*, pages 103–116. ACM, 2015.
- [236] MAEDE ZOLANVARI. Emerging mimo technologies: Distributed, cooperative, massive, 3d, and full dimension mimo, April 2016. <https://www.cse.wustl.edu/~jain/cse574-16/ftp/mimo/index.html>.
- [237] BARRET ZOPH AND QUOC V. LE. Neural architecture search with reinforcement learning. In *arXiv:1611.01578*, 2017.
- [238] YONGPAN ZOU, WEIFENG LIU, KAISHUN WU, AND LIONEL M. NI. Wi-fi radar: Recognizing human behavior with commodity wi-fi. *IEEE Communications Magazine*, 55(10):105–111, Oct 2017.
- [239] YONGPAN ZOU, YUXI WANG, SHUFENG YE, KAISHUN WU, AND LIONEL M. NI. Tagfree: Passive object differentiation via physical layer radiometric signatures. In *2017 IEEE International Conference on Pervasive Computing and Communications (PerCom)*, pages 237–246, March 2017.