The Pennsylvania State University
The Graduate School

FINGER MOTION ANALYTICS FOR INTERACTIVE APPLICATIONS USING WEARABLE AND WIRELESS IOT DEVICES

A Dissertation in
Computer Science and Engineering
by
Yilin Liu

Submitted in Partial Fulfillment
of the Requirements
for the Degree of

Doctor of Philosophy

May 2024

The dissertation of Yilin Liu was reviewed and approved by the following:

Mahanth Gowda
Assistant Professor, Department of Computer Science Engineering
Dissertation Advisor
Chair of Committee

Bhuvan Urgaonkar
Professor, Department of Computer Science Engineering

Bin Li
Associate Professor, Department of Electrical Engineering

Amulya Yadav
Assistant Professor, Department of Information Sciences and Technology

Chitaranjan Das
Distinguished Professor, Department of Computer Science Engineering
Head of the Department

Abstract

In this dissertation, I delve into innovative finger tracking methods, underscoring their potential applications across a variety of fields. My exploration begins with the FinGTrAC system, designed to recognize discrete finger gestures for American Sign Language (ASL) translation using minimal hardware: a single sensor on a finger and a smartwatch. Subsequently, I introduce NeuroPose, a system for continuous 3D finger motion tracking using wearable ElectroMyoGraphy (EMG) sensors, combining anatomical constraints with machine learning techniques. This technology, when extended via a mirrored bilateral training scheme, demonstrates potential applications for prosthetics. Addressing the pressing need for large-scale training data in machine learning, I propose ZeroNet, a system that uses publicly available video data for inferences on Inertial Measurement Unit (IMU) sensors, delivering high recognition accuracy. Lastly, I explore less intrusive tracking solutions with mm4Arm, a system that employs millimeter wave (mmWave) sensors to track 3D finger motion by focusing on forearm reflections correlated with finger movements. Through these investigations, my research provides a comprehensive understanding of finger tracking technologies' opportunities and challenges, paving the way for their integration into the rapidly evolving Internet of Things (IoT) landscape.

Table of Contents

List of Figures

List of Tables

Acknowledgments

I would like to express my sincere gratitude to a number of individuals whose support and guidance have been invaluable throughout my PhD journey.

First and foremost, I extend my deepest thanks to my PhD adviser, Dr. Mahanth Gowda, whose expertise, understanding, and patience, added considerably to my graduate experience. Your guidance helped me navigate through the challenges of research and writing, and your encouragement made all the difference in these years.

I am also immensely grateful to my committee members, Dr. Bhuvan Urgaonkar, Dr. Bin Li and Dr. Amulya Yadav, for their insightful feedback and unwavering support throughout this process. A special word of thanks to Dr. Srihari Nelakuditi for the valuable instruction and collaboration on the mm4Arm project, which was indeed helpful in shaping my research direction and methodology.

My time was enriched by the opportunities of internships under the mentorship of Dr.Ramanujan Sheshadri from NEC Labs, Dr. Bing Zhou from Snap, Inc , and Dr. Boyan Bonev from Google. The experience and knowledge gained during these internships were invaluable, and I am thankful for the guidance, encouragement, and opportunities provided by my mentors and managers at these esteemed organizations.

I owe a debt of gratitude to my family, whose love, support, and sacrifices have been the bedrock of my strength and perseverance. Your unwavering belief in my abilities has been a source of constant motivation. To my girlfriend, thank you for your understanding, patience, and endless love. Your support during this academic endeavor has been a source of peace and happiness.

Last but certainly not least, I want to acknowledge my lab mates for their collaboration, camaraderie, and shared insights. Working alongside you has been one of the most enjoyable aspects of my PhD experience. Your support, friendship, and shared enthusiasm for research have made my time in the lab both productive and enjoyable.

This dissertation stands as a testament to the collaborative effort and mutual support of this incredible community. Thank you all for your contributions to my journey.

Chapters 1, 2, and 3 contain materials from "Finger gesture tracking for interactive applications: A pilot study with sign languages", by Yilin Liu, Fengyang Jiang and Mahanth Gowda, in Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies. The author of this dissertation is the primary investigator of this paper. Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that

This document is based upon work supported by the National Science Foundation under Award No. 1956276, 2008384, 1909479, and 2046972. Any opinions, findings, and conclusions or recommendations expressed in this document are those of the author and do not necessarily reflect the views of the National Science Foundation.

Chapter 1
Introduction

Nowadays, finger tracking enables a number of exciting applications in sports analytics [13], healthcare and rehabilitation [14], sign languages [15], augmented reality (AR), virtual reality (VR), haptics [16] etc. Analysis of finger motion of aspiring players can be compared to experts to provide automated coaching support. Finger motion stability patterns are known to be bio-markers for predicting motor neuron diseases [17]. Finger gestures for sign language translation can significantly help deaf and hard of hearing (DHH) in communicating with people with normal hearing ability thus motivating more than two decades of research [18, 19]. AR/VR gaming as well as precise control of robotic prosthetic devices are some of the other applications that benefit from finger tracking [20, 21].

I get started from discrete finger gestures, more specifically, exploring the feasibility of finger gesture tracking in the application of American Sign Language (ASL) translation. Five fingers of a human hand posses more than 30 degrees of freedom. Tracking these degrees of freedom to detect motion of hand gestures might ideally require multiple sensors to be placed on the hand, which can be intrusive. Our system *FinGTrAC* explores the limits of feasibility of finger gesture tracking using a single sensor placed on the finger as a ring and a smartwatch worn on the wrist. The key contribution of this work is in scaling up gesture recognition to hundreds of gestures while using only a sparse wearable sensor set where prior works have been able to only detect tens of hand gestures. *FinGTrAC* exploits a number of opportunities in data preprocessing, filtering, pattern matching, context of an ASL sentence to systematically fuse the available sensory information into a Bayesian filtering framework. Culminating into the design of a Hidden Markov Model, a Viterbi decoding scheme is designed to detect finger gestures and the corresponding ASL sentences in real time. Extensive evaluation on 10 users shows a recognition accuracy of 94.2% for 100 most frequently used ASL finger gestures over different sentences.

Given the discrete finger gesture recognition's high accuracy, the idea of 3D continuous

hand pose tracking came to my mind. I designed a system called *NeuroPose*, that hows the feasibility of 3D finger motion tracking using a platform of wearable ElectroMyoGraphy (EMG) sensors. EMG sensors can sense electrical potential from muscles due to finger activation, thus offering rich information for fine-grained finger motion sensing. However converting the sensor information to 3D finger poses is non-trivial since signals from multiple fingers superimpose at the sensor in complex patterns. Towards solving this problem, *NeuroPose* fuses information from anatomical constraints of finger motion with machine learning architectures on Recurrent Neural Networks (RNN), Encoder-Decoder Networks, and ResNets to extract 3D finger motion from noisy EMG data. Furthermore, a transfer learning algorithm is leveraged to adapt a pretrained model on one user to a new user with minimal training overhead. A systematic study with 12 users demonstrates a median error of 6.24° and a 90%-ile error of 18.33° in tracking 3D finger joint angles.

After exploring EMG applications like *NeuroPose*, I found that an important application of EMG devices is in developing prosthetic devices for amputees with missing fingers. However, because of missing fingers, it is non-trivial to generate training data that maps EMG signal pattern into corresponding 3D joint angles of various fingers. Towards handling this challenge, I explore a *mirrored bilateral training* [22] scheme and validates the feasibility of applying mirrored bilateral training approach in prosthetic devices, which is an extended version of *NeuroPose*.

NeuroPose do requires a large amount of training data, and so do other machine learning (ML) based systems. While finger motion tracking with cameras is very mature, largely due to availability of massive training datasets, there is a dearth of training data for developing robust ML models for wearable IoT devices with Inertial Measurement Unit (IMU) sensors. Therefore could we have a zero-training overhead finger tracking system using wearables? Yes - Our system *ZeroNet*, which harvests training data from publicly available videos for performing inferences on IMU. The difference in data among video and IMU domains introduces a number of challenges due to differences in sensor-camera coordinate systems, body sizes of users, speed/orientation changes during gesturing, sensor position variations etc. *ZeroNet* addresses these challenges by systematically extracting motion data from videos and transforming them into acceleration and orientation information measured by IMU sensors. Furthermore, data-augmentation techniques are exploited that create synthetic variations in the harvested training data to enhance the generalizability and robustness of the ML models to user diversity. Evaluation with 10 users demonstrates a top-1 accuracy of 82.4% and a top-3 accuracy of 94.8% for recognition of 50 finger gestures thus indicating promise.

While wearable sensor performs well in tracking the fingers, it still needs on-body sensors

which users might feel uncomfortable and intrusive. Therefore, I explore finger tracking solutions using RF and wireless sensing techniques. More specifically, I choose mmWave sensors due to its high resolution and good penetrability. mmWave signals form a critical component of 5G and next-generation wireless networks, which are also being increasingly considered for sensing the environment around us to enable ubiquitous IoT applications. In this context, I propose a work that leverages the properties of mmWave signals for tracking 3D finger motion for interactive IoT applications. While conventional vision-based solutions break down under poor lighting, occlusions, and also suffer from privacy concerns, mmWave signals work under typical occlusions and non-line-of-sight conditions, while being privacy-preserving. In contrast to prior works on mmWave sensing that focus on predefined gesture classification, this work performs continuous 3D finger motion tracking. Towards this end, I first observe via simulations and experiments that the small size of fingers coupled with specular reflections do not yield stable mmWave reflections. However, I make an interesting observation that focusing on the forearm instead of the fingers can provide stable reflections for 3D finger motion tracking. Muscles that activate the fingers extend through the forearm, whose motion manifests as vibrations on the forearm. By analyzing the variation in phases of reflected mmWave signals from the forearm, this paper designs mm4Arm, a system that tracks 3D finger motion. Nontrivial challenges arise due to the high dimensional search space, complex vibration patterns, diversity across users, hardware noise, etc. mm4Arm exploits anatomical constraints in finger motions and fuses them with machine learning architectures based on encoder-decoder and ResNets in enabling accurate tracking. A systematic performance evaluation with 10 users demonstrates a median error of 5.73° (location error of 4.07 mm) with robustness to multipath and natural variation in hand position/orientation. The accuracy is also consistent under non-line-of-sight conditions and clothing that might occlude the forearm. mm4Arm runs on smartphones with a latency of $19ms$ and low energy overhead.

The rest of the thesis is structured as follows. Chapter 2 provides some related works of finger tracking areas. Chapter 3 introduces *FinGTrAC* [23, 24] – Fine-grained finger gesture tracking system using low intrusive wearable sensor platform. Chapter 4 introduces *NeuroPose* [25] – 3D hand pose continuous tracking system using EMG wearables. Chapter 5 introduces extended version of *NeuroPose* [26] being applied with *mirrored bilateral training*– 3D Hand Pose Tracking using EMG Wearables with Applications to Prosthetics. Chapter 6 introduces *ZeroNet* [27] – Zero training overhead finger tracking system using IMU sensor. Chapter 7 introduces *mm4Arm* [28] – Leveraging the Properties of mmWave Signals for 3D Finger Motion Tracking for Interactive IoT Applications.

Chapter 2
Related Works

2.1 Vision

Finger motion can be captured by depth cameras like kinect [29] and leap [30] sensors. However, advances in machine learning, availability of large training datasets have enabled precise tracking of finger motion even from monocular videos that do not contain depth information [31, 32]. While such works are truly transformative, we believe wearable based solutions have benefits over vision based approaches which are susceptible to occlusions, lighting, and resolution. In addition, wearable devices offer ubiquitous solution with continuous tracking without the need of an externally mounted camera. FingerTrak [33] has innovatively designed wearable thermal cameras to track 3D finger motion. However, tracking may not be robust under changes in background temperature as well as motion of wrist (due to shift in camera positions).

2.2 Sensor Gloves

Gloves with embedded sensors such as IMU, flex sensors, and capacitative sensors have been used for finger pose tracking in a applications like sign language translation, gaming, HCI etc [15]. Work in [34] tracks hand pose using an array of 44 stretch sensors. Works [35, 36] extracts hand pose using gloves embedded with 17 IMU. Flex sensors have been used in commercially available products such as CyberGlove [37], ManusVRGlove [38], 5DT Glove [39] etc. However, wearing gloves in hands may hinder dexterous/natural hand movements [40].

2.3 Radio Frequency (RF)

RF signals have been used for human body motion sensing [41–44]. They are also used to track the motion of the hand and classify discrete gestures by using a combination of wireless channel state information (CSI), and Doppler shifts [45–47]. Wireless signals have been used for a number of applications [48–50]. WiSee [51] can detect hand gestures by measuring doppler shifts from WiFi reflections. mmWrite [52] performs handwriting recognition using mmWave radars. RFWash [53] detects hand wash hygiene using mmWave radars near bathroom mirrors. SignFi [54] uses CSI from WiFi APs for sign language recognition. ExASL [55] tracks point clouds computed from range-doppler spectrum and angle of arrival spectrum of mmWave radars. This is used to classify upto 23 discrete gestures used in ASL. 3D pose of the human body has been detected even behind occlusions such as Walls using wireless body reflections [56, 57]. Heart rate, breathing, and physiological signals of interest to healthcare applications have been detected using RF signals [58, 59]. Google project Soli [60] can detect fine grained finger gestures using mmWave reflections. RF based tracking requires the need for any external infrastructure, like vision, but it is completely passive.

2.4 Inertial Sensors

Inertial sensors have been used in many localization and gesture tracking applications [61, 62]. UnLoc [63] fuses information from smartphone sensors for extracting characteristic fingerprints in indoor environments for localization. RisQ [64] recognizes smoking gestures for appropriate intervention measures using smartwatches. Similarly, smartwatches are used for eating activity recognition [65] and measuring calorie intakes. DUI [66] detects blood alcohol level based on user performance on smartphone activities. Other applications have been explored in the areas of augmented and virtual reality, sports analytics, smart-health, and security [67–70].

2.5 ElectroMyoGraphy

EMG based gesture tracking is an active area with decades of research. Prior works perform classification of discrete hand poses [71–75] or tracking of a predefined sequence of hand poses [74, 75] using EMG sensors with a combination of deep learning techniques based on CNN, RNN etc. Work in [76] can classify multi finger gesture sequences using a 4 channel EMG sensor. A number of popular features based on spectral power magnitudes, hudgins' time domain features, correlation coefficients etc have been used in conjunction with SVMs, nearest

neighbors, and linear discriminant analysis based algorithms to show the feasibility of gesture classification. Work in [77] uses Myo armband similar to the one used in this paper to classify 5 gestures such as fist, wave-in, wave-out, open, and pinch etc. A shallow feed forward neural network with 3 layers has been used to perform this classification. Work in [78] shows that muscle synergy can be exploited to reduce the dimensions of feature vectors in EMG based gesture classification. Evaluated over five hand activities such as open, close, pinch, valgus, and grasp, the recorded EMG data from the forearm have been compressed using non negative matrix factorization to extract synergistic myo-electrical activities. The compressed feature set has shown to demonstrate a higher recognition rate. Work in [79] uses forearm EMG signals to control a robotic arm. A set of 9 gestures are detected to contral a 6 degree of freedom robotic arm. Ensembled bagged trees, SVM, and neural networks have been used to perform the classification. Works [80] can track joint angles for arbitrary finger motion, but requires a large array of more than 50 EMG sensors placed over the entire arm. Work in [81] tracks joint angles using EMG sensors but only for one finger.

2.6 Mirrored bilateral training

An important application of EMG devices is in developing prosthetic devices for amputees with missing fingers. Work in [22] estimates the force on contralateral arm using EMG signals measured from the other arm. A multilayer perceptron (MLP) based algorithm has been used to make the association between EMG signals and the associated force in the arm. Based on several experiments with tens of individuals , this paper shows that an accurate estimation of forces in the contralateral limb can be done based on the EMG signal from the other arm, thus showing promise. Similarly, work in [81] shows the feasibility of estimation of flex and extension joint angles of one finger based on EMG data collected from the other hand. A number of features such as zero crossings, mean absolute value, waveform length, slope changes etc has been applied on EMG data. Furthermore, a state space model with parameters estimated from contralateral arm is used to estimate the joint angle of one finger on the other arm. The results show an estimation error under 1 degree thus indicating sufficient promise. Work in [82] compares training via mirrored EMG from contralateral arm with training by mimicking gestures on the same arm with potential amputation. Evaluated over more than 20 gestures, a better performance is achieved by mirroring on the contralateral arm instead of mimicking with the same arm that may have amputation. The main challenge with mimicking is identified as the inability to estimate force involved in motion as well as misalignment over time with between the imitation and the actual gesture. Work in [83] can perform wrist motion

classification using *mirrored bilateral training*. Based on the EMG data from the contralateral arm, and employing techniques based on artificial neural networks for pattern classification, upto 70% in accuracy has been shown in terms of classification of 4-6 wrist motion gestures. All of the above works show promise in the technique of *mirrored bilateral training*.

2.7 Machine Learning Strategy

In the realm of on-device machine learning, significant strides have been made towards efficient computation on low-power and intermittently powered systems [84–89]. Data augmentation enriches the quality of datasets to help ML models generalize well and exhibit higher accuracy and robustness with limited quantity of training data. Transformation such as rotation, scaling, translation and elastic distortions on images have been explored to create more training data from existing datasets. [90–93]. Similarly, image cropping, flipping, color shifting, and whitening are other techniques to create new training data from existing datasets [94]. In the area of automatic speech recognition (ASR), data augmentation techniques such as frequency axis distortions [95], speech rate variations, vocal tract normalization [96] etc have been explored to improve the accuracy. This is particularly important in the context of IMU data since there is no large scale public datasets like computer vision or speech. Data augmentation techniques have been explored in the context of wearable sensing for parkinson disease gait monitoring [97] and construction activity monitoring [98]. More recently, data augmentation for human activity recognition has been extensively studied in [99] for several benefits including robustness to sensor wearing positions.

2.8 Transfer Learning from Videos

Deep Inertial Poser [100] uses synthetic data from motion capture videos (from cameras like ViCON [101]) instead of public videos for training human pose tracking algorithms with 6 on-body IMUs. Such motion capture cameras can provide high quality training data with mm level accuracy. However, creating such datasets requires 6-8 costly ViCON cameras. We believe using publicly available videos is an easier alternative. More recently, several innovative works [102–106] have explored the use of YouTube-like videos for training human activity recognition (HAR) on wearable sensors.

Chapter 3
FinGTrAC

3.1 Introduction

Five fingers of a human hand posses more than 30 degrees of freedom. Tracking these degrees of freedom to detect motion of hand gestures might ideally require multiple sensors to be placed on the hand, which can be intrusive. This paper explores the limits of feasibility of finger gesture tracking using a single sensor placed on the finger as a ring and a smartwatch worn on the wrist. While such sparse sensors may not sufficiently capture all degrees of freedom, we present a system called FinGTrAC (**Fin**ger **G**esture **Tr**acking with **A**pplication **C**ontext) that exploits application specific context to fill in for the missing sensor data. For example, basketball players maintain a specific wrist angle, careful flexing of finger joints and an optimal positioning of index and middle fingers before shooting the ball [107]. Virtual reality applications have similar prior probabilities of finger configurations. We argue that such application specific context can fill the gap in sensing, and track the main finger motion metrics of interest. While prior works [108–113] on wearable finger gesture tracking that use non-intrusive sensors are only limited to recognition of tens of gestures, we show how a

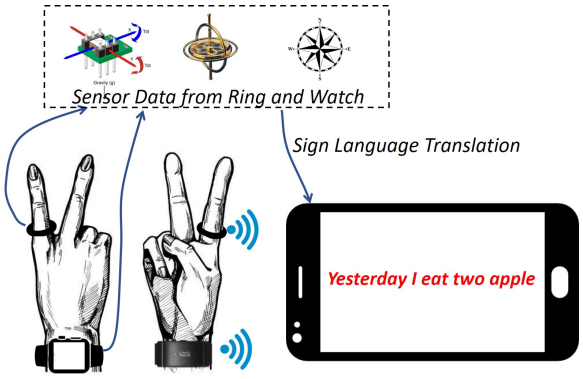


Figure 3.1: ASL recognition with smart devices [1]

similar setup can be scaled to recognition of hundreds of gestures by exploiting application specific context. Although we do not validate any generic claim in this paper over different applications, we make our case with an example application in American Sign Language (ASL) translation. A sign language is a way of communication that uses body motion (arms, hands, fingers) and facial expressions to communicate a sentence instead of auditory speech. We show the feasibility of translating sentences composed of 100 most frequently [114] used ASL words. While the sensor data is under-constrained, we fuse them with Bayesian inferencing techniques that exploit the context information in a ASL sentence towards achieving a higher accuracy.

Finger motion tracking has a number of important applications in user-interfaces and creating accessibility for patients with motor related disorders. In augmented reality, finger motion tracking enables the richness of user interaction with augmented objects. In sports, finger motion data can be used for coaching players as well as managing injuries. In smart-health, fine-grained finger motion data analysis can reveal digital biomarkers for various motor related diseases. Finger gestures for ASL translation – the subject of this paper – can significantly help deaf and hard of hearing (DHH) in communicating with people with normal hearing ability thus motivating more than two decades of research [18, 19]. The DHH population is about 10 million [115] in US, 466 million globally and estimated to be 900 million by 2050 [116].

Prior works can be broadly classified into wearable based approaches and camera based approaches. Wearable based approaches are only limited to a few tens of gestures [108, 111–113] or use intrusive setup of sensor gloves (15-20 sensors) and Electromyography (EMG) electrodes for detecting upto hundred gestures [15, 117]. While cameras [118] track full finger motion, the accuracy depends on lighting/resolution as well as the presence of user in the camera view. Wearable based approaches in general offer more ubiquitous solution than cameras. An innovative work, SignSpeaker [119] performs translation of 103 ASL gestures with smartwatches using deep learning approaches [120] popular in speech recognition. Evaluated in Section 3.7, *FinGTrAC's* primary distinction lies in detecting any unseen sentence composed from a ASL dictionary, whereas SignSpeaker cannot detect unseen sentences outside the training database. Given the impracticality to train over infinitely possible sentences in a language, we believe FinGTrAC has non-trivial benefits over SignSpeaker.

In contrast to prior works, *FinGTrAC* differs in following ways: (1) *FinGTrAC* exploits application specific context in ASL for finger motion tracking and scaling gesture recognition from few tens to hundreds of gestures. (2) *FinGTrAC* uses a single sensor on one finger instead of using additional sensors on all fingers. (3) In addition to tracking hand motion during word gestures, *FinGTrAC* also tracks hand motion in between words of a sentence for higher accuracy. (4) *FinGTrAC* can detect arbitrarily new sentences without any training. (5) A minimal training

at the level of words (Section 3.5) by a single user is sufficient, and this model extends to any new user. Our recent work [1] presents a proof-of-concept presentation of this idea with limited evaluation. In contrast, this paper performs a thorough evaluation which includes performance characterization with different sensor placements, accuracy variation across users, comparison with state of the art approaches such as [119], potential to improvement in accuracy with human assistance etc. In addition, new techniques for finger-spelling detection are proposed and validated. The techniques proposed in the paper have been expanded with characterization of intermediate results. The raw sensor data is presented with preliminary analysis.

Fig. 3.1 depicts the system. A user would wear a smart ring on the index finger (index finger is the most involved finger in ASL finger gestures), and a smartwatch on the wrist. Since most people are comfortable wearing one ring and a watch in daily life, we chose this platform for the study. The Inertial Measurement Unit (IMU) sensors (accelerometers, gyroscopes, and magnetometers) on the smart ring and smart watch are used for finger tracking, and the results might be displayed on smartphone screen, or read out on speaker. A number of non-trivial challenges emerge as enumerated below. (1) The data is severely under-constrained. All fingers and both hands are involved in finger gestures, whereas *FinGTrAC* uses a sensor only on the index finger of the dominant hand. (2) The sensor data is noisy. Because of this subtle differences in ASL gestures cannot be captured accurately. (3) A huge diversity in gesturing exists across different users. *FinGTrAC* needs to maintain high accuracy and robustness despite such variations.

Towards addressing these challenges, *FinGTrAC* exploits a number of opportunities as enumerated below. (1) A ring on the index finger cannot sufficiently capture all degrees of motion of the hand to detect ASL words. Therefore, in addition to tracking finger motion during the sign gesture for a particular word, we also track hand motion in between words of a sentence. This provides a lot of contextual information. This opens up opportunities for higher accuracy gesture detection in the context of an ASL sentence than in isolation. (2) Extraction of high level motion features combined with techniques such as Dynamic Time Warping (DTW) can narrow down the search space for words while simultaneously increasing the robustness across diverse users. (3) The knowledge of the dictionary from which words are drawn can dramatically improve the accuracy of detection of finger-spelling signs. (4) Decoding words in a sentence can be modeled as a Bayesian Filtering problem (HMM) analogous to packet decoding over a noisy wireless channel. Thus, we can leverage Viterbi decoding to provide optimal sentence decoding that makes best use of the noisy sensor data.

FinGTrAC is implemented on a platform consisting of: (1) An off-the-shelf button shaped sensor (VMU931 [121]) worn on the index finger like a ring. (2) SONY Smartwatch 3 SWR50

worn on the wrist. The sensor data from the watch and ring is streamed continuously to an edge device (desktop) which decodes sentences in real time (0.4 seconds even for 10 word sentences). With a study from 10 users, *FinGTrAC* achieves an accuracy of 94.2% in detecting words from 50 sentences composed from 100 most commonly used ASL words. Furthermore, scalability analysis shows that extending the dictionary size to 200 words offers a graceful degradation in accuracy.

To our best knowledge, *FinGTrAC* outperforms other wearable ASL solutions that use 15-20 sensors [15, 117]. On the other hand, in contrast to low intrusive setup that can track few tens of gestures, FinGTrAC scales recognition to hundreds of gestures. Besides what is achievable, we also discuss shortcomings. Our goal is to show the feasibility of finger motion tracking by exploiting application specific opportunities, and we use ASL as a case-study with a dictionary of 100 most frequently used ASL words. However, we note that ASL has a larger vocabulary, facial expressions, and a rich grammar. The complexity is comparable to spoken languages. Hence, full ASL translation is outside the scope of our paper, but we believe we make a significant first step. We discuss opportunities in sensing, machine learning and natural language processing towards extending this work for full ASL translation (Section 3.9). Considering this, our contributions are enumerated next:

(1) To the best of our knowledge, FinGTrAC is the first system that shows the limits and feasibility of using a single smart ring and a smart watch (low-intrusive) for finger motion tracking with application specific context from ASL.

(2) FinGTrAC scales gesture recognition to hundreds of gestures in contrast to tens of gestures in prior work but only requires a low fidelity training from a single user (user independent training). In the context of ASL, the recognition generalizes to unseen new sentences composed from a given dictionary of words.

(3) FinGTrAC systematically combines information from the sensors in the context of an ASL sentence into a Bayesian inferencing model for high accuracy sentence decoding.

(4) Implementation is done on user-friendly platforms of smart-ring and smart-watch for detection in real time.

(5) A study with 10 users on 50 sentences from a dictionary of 100 most popular words shows an accuracy of 94.2% .

3.2 Background: Application Domain

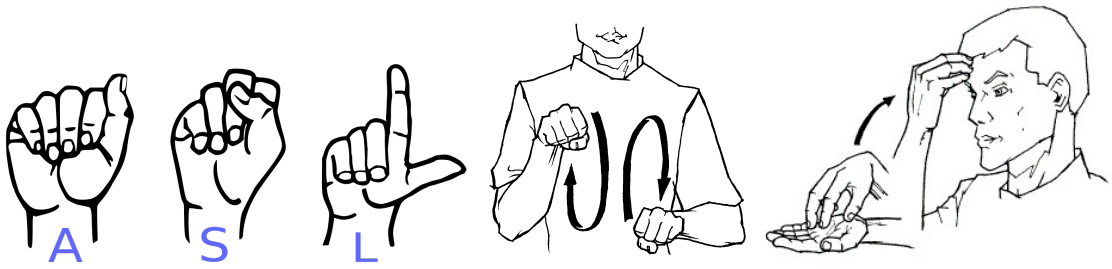
We begin with a brief overview of sign languages, hand gestures, and ASL grammar.

Signing and Gestures: Sign languages use gestures instead of sound for communication.

Table 3.1: ASL sentence vs English sentence

ASL Sentence	English Sentence
I Drink Water	I need to drink water
My Mother Sleep Home	My mother is sleeping at home
Ball, Boy Throw	The ball was thrown by the boy
Bathroom where?	Where is the bathroom?

Sign languages including ASL are considered a class of natural languages with their own grammar and lexicon. Approximately, there are 137 sign languages with millions of speakers. ASL is primarily used by the in USA and parts of Canada. Majority of ASL signs involve the motion of the dominant hand including fingers. Fig. 3.2(a) shows the hand and finger poses for signing the letters - A, S and L. Fig. 3.2(b),(c) shows hand motions involved in signing "bike" and "learn". As shown in Figure, the non-dominant hand can also be a part of ASL signs. The non-dominant hand and facial expressions are used occasionally to complement the dominant hand gestures. For example, eyebrows are raised at the end of a sentence to ask a yes/no question. ASL signs include words, 26 alphabets, and 10 digit signs [122]. The vocabulary of an ASL user is a few thousand words. Finger-spellings are used to sign words for which no direct ASL sign exist (ex: "sensor" does not have an ASL sign). After a new word is introduced by finger-spelling, a reference is created in space for the new word. Subsequent communications that reuse the same word can simply point the hands or gesture towards the spatial reference without spelling the word each time.



((a)) ASL Finger-spelling ((b)) ASL sign: Bike ((c)) ASL sign: Learn

Figure 3.2: Examples of hand poses and motion in ASL

ASL Grammar: ASL grammar mainly consists of sentences in the following format: *Subject-Verb-Object* [123]. Sometimes, the word order can change when the context of the topic is established first through topicalization [124]. This results in the following format: *Object, Subject-Verb*. Table 3.1 shows some example ASL sentences and corresponding English translations. Facial expressions are used to denote emphasis, questions etc. Pauses indicate end of sentences and words [125]. ASL does not have a copula [124] such as "is" to connect

Figure 3.3: ring, watch platform

subject and predicate and it does not use BE verbs (am, is, are, was, were). Also, the signs do not indicate tense of a verb such as *washed* or *washing*. Therefore, the following format is often used to resolve ambiguity [126]: Time-Subject-Verb-Object. For example, WEEK-PAST I WASH MY CAR is the equivalent to saying *I washed my car last week* in English with the appropriate time and tense. Similarly, NONE/NOT is introduced at end of sentences to indicate negation. For example, TENNIS I LIKE PLAY NOT is the ASL equivalent to *I don't like to play tennis*.

3.3 Platform Description

Our ideal platform consists of a smart-ring worn on a finger and a smart-watch worn on the wrist. The ring is worn on the index finger since the index finger is more frequently involved in gestures. Smart rings that can pair with phones wirelessly to stream information as well a monitor activity are already available on the market [127, 128]. For example, the oura ring [128] is popular as a sleep tracking device and weighs between 4 and 6 grams, which is even lighter than conventional rings, and packaged in a stylish design. It is low intrusive with users finding it comfortable for wearing day and night, gym, pool etc [129], thus receiving favorable online reviews for usability [129–131]. However, most of these platforms are closed and do not provide access to raw sensor data. Therefore we use a button-shaped sensor – VMU931 [121] snugly fit on the finger like a ring as shown in Fig. 3.3. Given that VMU931 does not support wireless streaming, we connect it with a USB cable to a Raspberry PI, which streams data to a desktop over USB WiFi. The ring (VMU931) and watch (SONY SmartWatch 3 SWR50), both generate 9 axis IMU data during ASL finger gestures - 3 axes each for Accelerometer, Magnetometer, and Gyroscope. This forms the input to NeuroPose. Hand motion is tracked

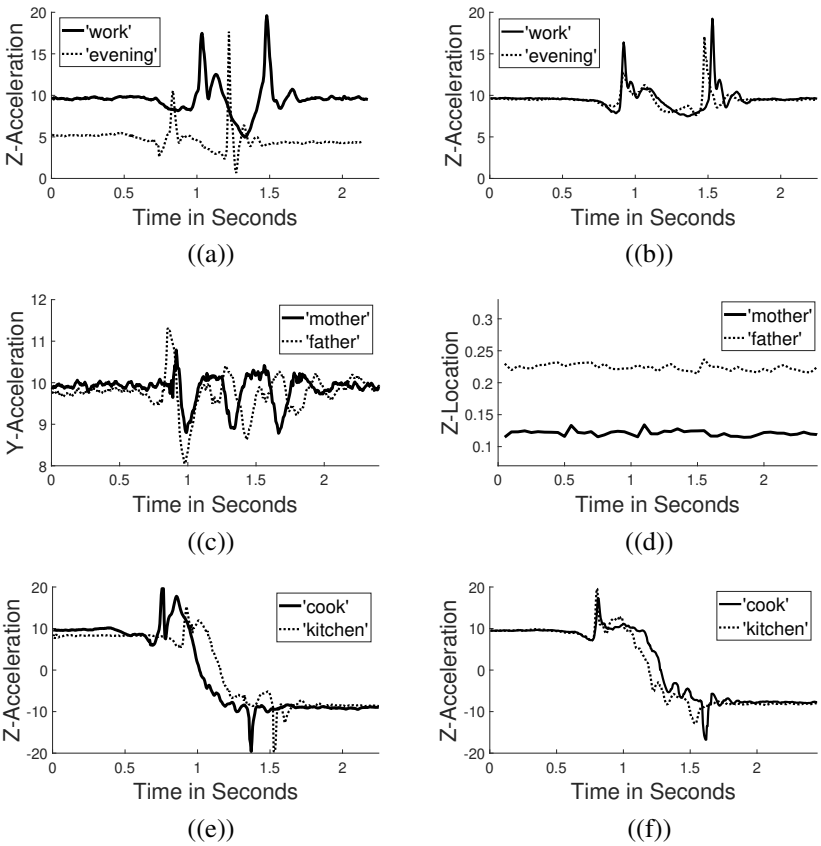


Figure 3.4: Significance/limitations of ring and watch data in ASL gesture detection. The first and second columns contain ring and watch data respectively. The three rows provide a comparison of ring and watch data over three different word pairs.

by the watch to obtain wrist and elbow locations using *MUSE* [132]. The ring captures subtle finger motions. Since we do not have sensors on all fingers, the sensor data does not sufficiently capture the entire ASL gesture. We now look at the raw data for special cases to discuss feasibility.

3.4 Raw Sensor Data: A first look

Figure 3.4 shows the raw sensor data from watch and ring for a few interesting cases. Apart from raw IMU data from ring, and watch, NeuroPose uses wrist location derived from the watch [132] for gesture detection. Tracking finger location is beyond the scope of this project (Future possibilities discussed in Section 3.9).

We include these examples to point out a few observations: (1) The smartwatch data for gestures of words "work" and "evening" is shown in Fig. 3.4(b). These two words have identical wrist locations and orientations, and their motion data look very similar to each other.

Although only z-axis data of the accelerometer is shown for clarity, the data from other axes as well as the data from gyroscope and magnetometer are also similar. Thus the two words cannot be distinguished by watch data alone. However, the words have different finger motions which are accurately captured by the ring sensor data in Fig. 3.4(a) (The figure captures differences in finger orientation across these two gestures). Thus the ring sensor data is critical in capturing subtle finger motions considering that many word pairs in ASL (such as "Friday" and "Saturday"; "I" and "my"; finger-spellings etc) have similar wrist orientations and location. (2) For words such as "mother" and "father", the ring data is very similar (Figure 3.4(c)) because the words have identical index finger motion. To resolve ambiguity in such cases, NeuroPose tracks wrist location differences (Figure 3.4(d)) across these words and incorporates the opportunity into a Bayesian filtering framework. Other challenges arise due to variation in gestures across users, noisy data etc which are addressed in Section 3.5. Nevertheless, a combination watch and ring data looks viable to classify most ASL gestures. (3) Although rare, for words such as "cook" and "kitchen", both watch and ring generate very similar data (Figures 3.4 (e) and (f)) because both the hand poses and index finger motions are identical. NeuroPose cannot handle such cases currently, but in Section 3.9, we argue how Natural Language Processing (NLP) [133] techniques can potentially resolve such ambiguities in the future.

3.5 Core Technical Modules

3.5.1 Data Segmentation

An ASL sentence is a sequence of words. Given a stream of IMU sensor data, we first segment it into alternative blocks of the following two key phases: (1) "word phases", where a user is signing word and (2) "transition phases", where the hand is moving to the new location to sign a new word. The hand pauses briefly at the beginning and end of each word [125]. We use this opportunity to segment the sensor data.

Fig. 3.5(a) shows the gyroscope data (magnitude) from the ring for the following sentence: "You full understand?". The data is labelled with corresponding words. "T" denotes "transition phases". Clearly, we can observe some characteristic *dips* in the measurements as pointed out in the figure. This corresponds to the micro-pauses during word transitions. Towards automatically detecting the *dips* for segmentation, we first derive a *dip template* that captures the properties of a *dip* as shown in the subplot in Fig. 3.5(a). Then, we perform a pattern matching with DTW [134] to search for the occurrence of the above *dip template* in the gyroscope sensor

data of each sentence. This provides a robust approach for detecting transitions even when the duration and nature of the pause signature varies across users and time.

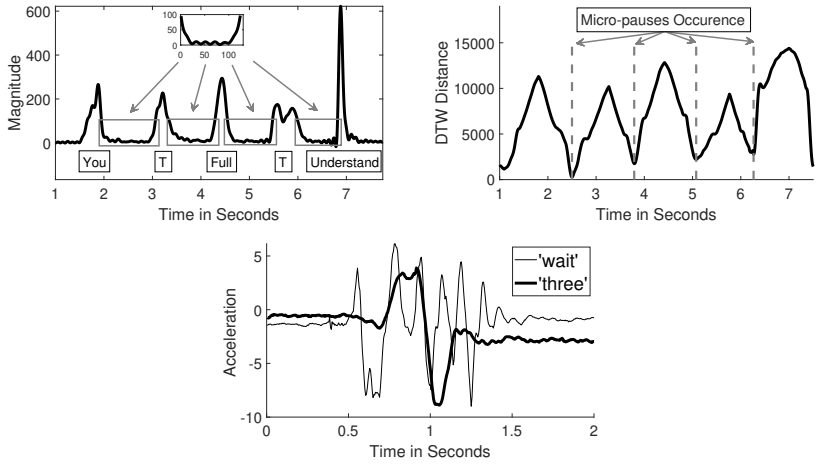


Figure 3.5: (a) Micro-pauses between words can be used for segmentation (b) Segmentation based on DTW matching of the *dip* template (c) "wait" and "three" have different number of peaks in the accelerometer data

Even with words that have minimal motion of the fingers or hand during "word-phases" or "transition-phases", we observe that there is always some signal either on the watch or the ring creating the *dip* templates. Fig. 3.5(b) shows the DTW distance of such a pattern matching. Clearly, there are local minimas corresponding to the locations of micro-pauses which can be used to reliably segment the sensor data into "word phases" and "transition phases".

3.5.2 Preprocessing

We preprocess the data with below steps to improve the robustness of word detection – particularly because different users may not perform the ASL signs in an exactly identical manner.

Low Pass Filtering: Human finger motions have low frequencies below 6 Hz even with rapid motions [135]. Similarly, hand and finger motions in ASL gestures are smooth and hence dominated by low frequencies too. Therefore, we first pass the sensor data through a low pass filter with a cutoff frequency of 10 Hz. This eliminates the noise and un-intended vibrations from higher frequencies and enhances the robustness of sign detection across multiple users with minimal training.

Elimination by Periodicity: The accelerometer data (magnitude) for signs "wait" and "three "is shown in Fig. 3.5(c). Sign for "wait" involves a periodic motion of the hand 5 times

and the data shows 5 peaks as expected. On the other hand, the periodicity pattern in "three" is different and shows only 1 peak as expected. We exploit the periodicity in narrowing down the search space. Consider matching an unknown gesture with 5 peaks with all words in the dictionary to find the best match. We first eliminate all words in the dictionary which do not have 5 peaks (such as the word "three"). This reduces the search space for the matching of the unknown word, thus improving the robustness and accuracy of further steps such as DTW. The significance of this step is quantified in the evaluation section 3.7.

3.5.3 Word Gesture Recognition and Ranking by DTW

We use Dynamic Time Warping (DTW) [134] to bootstrap word detection. Briefly, DTW is a pattern matching technique that inspects the overall shape of two signals to determine their similarity. For example, Fig. 3.6(a) shows the z-axis ring accelerometer data of two users gesturing the word "people". Although the overall shape is similar, parts of the motion traces

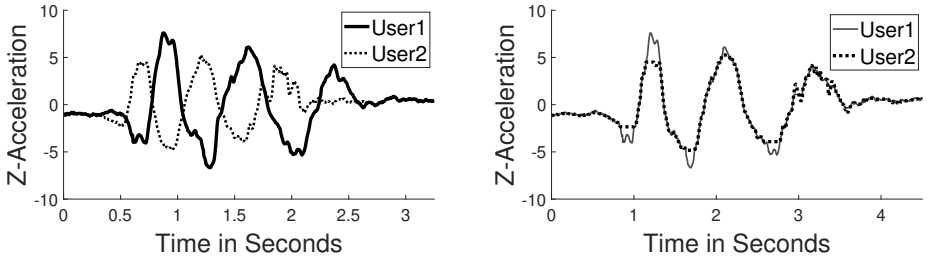


Figure 3.6: (a) Accelerometer data for "people" for two users (b) Data from user-2 is compressed and stretched to match with user-1 by DTW

happen at a faster rate or earlier for user-2 while other parts happen slower. DTW uses a dynamic programming optimization to minimally compress and stretch the two sequences relative to each other such that they provide the best overlap. Fig. 3.6(b) shows the two sequences after DTW optimization. DTW is known to do a good job of matching such series with similar overall shape. The residual differences between the two time series determines the similarity score between them.

Training: We first build a training database where labelled sensor data (9-axis IMU data) from the ring and watch is created for each ASL word. One user wears the prototype and signs all 100 ASL words in our dictionary. A one time training with a single user is sufficient, and no separate training is needed for each new user.

Word Gesture Detection: An unknown ASL gesture from a new user is matched using DTW with training data of each word in our ASL dictionary. The matching is done across all 9 dimensions of IMU data as well as the orientation estimates based on A3 [136]. The word

with the best match is most likely to be the unknown word. The ring data turned out to be more prominent in the word detection phase and the overall matching accuracy with DTW is 70.1%. This is because words-pairs such as "mother, father", "see, three" have similar wrist and index finger motions. Also, subtle variations in gesturing across users can cause miss-matches. To cope up with the poor accuracy of DTW matching, *FinGTrAC* considers not only the best match for an unknown word, but also the top 10 matches. The correct word is among top 10 ranks in 100% of the cases indicating promise.

The top 10 matches for each word from DTW are further processed with a Hidden Markov Model (HMM) to improve the accuracy. The HMM, particularly, incorporates wrist location transitions between words in the context of a sentence to decode the most likely sentence. Details are elaborated next.

3.5.4 Sentence Decoding with HMM and Viterbi

We use results from word detection as inputs to sentence detection algorithm. While results from word recognition might be inaccurate, we incorporate transition characteristics between words towards more accurate sentence detection. The HMM architecture for sentence detection is depicted in Fig. 3.7. In this section, we explain the details of the HMM.

Forward Pass: An ASL sentence is a sequence of words. *FinGTrAC* models *word*s within a sentence as the hidden *states* in HMM. The HMM first computes the likelihood probability of each dictionary word occurring as the t^{th} word in the sentence during a step called *Forward Pass*. Mathematically, we denote this probability as follows: $P(i|d_{1:t}) = L_i(t)$
In other words, given all sensor data from beginning of the sentence to the current word – $d_{1:t}$, $P(i|d_{1:t})$ denotes how likely is i^{th} dictionary word to occur as the t^{th} word in the sentence. These likelihood probabilities are computed from the below recursive equation.

$$L_i(t+1) = p(d_{t+1}|i) \sum_j L_j(t) p(i|j) \tag{3.1}$$

There are two key probabilities that the HMM model relies on while computing the above likelihood probabilities. First, $p(d_{t+1}|i)$ denotes the **emission probability**, which indicates how likely is the word i to generate the sensor observation during the $(t+1)^{th}$ "word phase" – d_{t+1}. Secondly, $p(i|j)$ is the **transition probability** which indicates how likely is a transition from word j to i happen in a sentence given the watch location difference between word transitions. The details are elaborated below.

■ *Emission Probability*: The emission probabilities $p(d_{t+1}|i)$ are derived from DTW

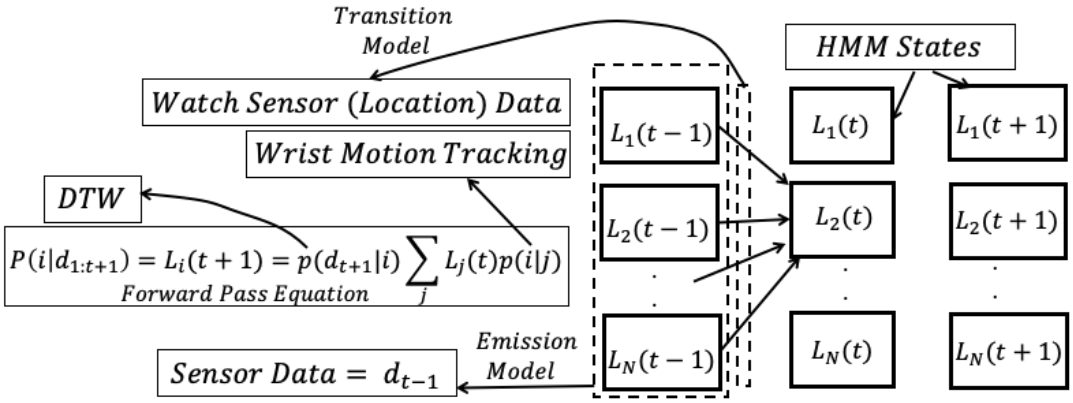


Figure 3.7: HMM model for sentence detection: Raw sensor data d_t contributes to the emission probability ($p(d_{t+1}|i)$) which is computed with DTW. Wrist location data contributes to transition probability- $p(i|j)$. Likelihood $L_i(t)$ of words at all parts of the sentence is first computed with forward pass. A backward pass with Viterbi algorithm decodes the most likely sentence.

matching.

$$p(d_{t+1}|i) \approx DTW\_metric(d_{t+1}, d_{i,template}) \tag{3.2}$$

As described in Section 3.5.3, the 9-axis IMU data for an unknown word is matched by DTW with labelled templates for each dictionary word $d_{i,template}$. The normalized similarity metric from DTW would be used to assign the *emission probabilities*. If we rank all words by similarity metrics, the correct word appears in the top 10 ranks in 100% cases. Thus we set probabilities of words beyond top 10 ranks to zero, to decrease the complexity of HMM.

■ *Transition Probability*: The transition probabilities are determined from the change in wrist location between two words as computed from the smart-watch.

We first determine wrist locations for each word in the dictionary using Kinect sensor. Kinect is only used for training data, and not a part of *FinGTrAC* system. This is a one time training overhead with a single user to determine the wrist locations for each word. Thus, Kinect locations at the start and end – $l_{st,kinect}(i)$ and $l_{ed,kinect}(i)$ for each word i in the dictionary is known with high precision. Later, during testing, a new user signs a sequence of unknown words forming a sentence. We compute wrist locations for each of these words by using techniques in MUSE [132] on the smartwatch data. Then, we update the transition probability as follows:

$$p(i|j) \propto \frac{e^{\frac{-i_d^2}{2\sigma^2}}}{\sqrt{2\pi\sigma^2}},$$

$$l_d = ||(l_{st}(i) - l_{ed}(j))| - |(l_{st,kinect}(i) - l_{ed,kinect}(j))||$$

$l_{ed}(j)$ denotes the end location of word j and $l_{st}(i)$ denotes the start location of word i as determined by the smartwatch data. σ^2 denotes the standard-deviation of location errors, which is approximately $10cm$.

■ *Prior Probability*: Prior knowledge about the states in HMM is useful in improving the accuracy of decoding. For example, if DTW matching metric is similar for "I" and "have", with all other information being equal, "I" is the more likely word signed by the user since "I" is used more frequently than "have" in ASL. We use the frequencies of our dictionary words [114] $f(i)$ as prior probabilities to improve the accuracy. Since the prior probabilities do not have dependency on previous states, they are similar in nature to emission probabilities. Thus, we integrate prior probabilities into emission probabilities and update the equation 3.2 to below:

$$p(d_{t+1}|i) \approx f(i).DTW\_metric(d_{t+1}, d_{i,template}) \tag{3.3}$$

At the end of forward pass, we have likelihood probabilities for each dictionary word at each position of the sentence. We next do a backward pass using Viterbi [137] algorithm to decode the most likely sentence.

Backward Pass: Viterbi Decoding: Consider a sentence of three words and two possible decodings. First decoding, say A estimates the words to be 34^{th},61^{st},4^{th} items of dictionary while second decoding, say B estimates the words to be - $12^{th}, 41^{th}, 3^{rd}$. Then, the following expressions denote the likelihood of the two decodings.

$$p(A) = p(d_1|34).p(61|34).p(d_2|61).p(4|61).p(d_3|4) \tag{3.4}$$

$$p(B) = p(d_1|12).p(41|12).p(d_2|41).p(3|41).p(d_3|3) \tag{3.5}$$

We can decide the more likely possibility among the two by comparing $p(A)$ and $p(B)$. We can extend this idea to compare all possible decodings and pick the one with highest probability. Viterbi algorithm [137] is an efficient way to do this under HMM assumption to decode the most likely sentence.

3.5.5 Finger-spelling Detection

English words that do not have a direct ASL sign are communicated by explicitly signing the characters forming the word. The hand and finger poses for various characters is depicted in Fig. 3.8. Identifying characters signed by a user is non-trivial because: (1) As seen in Fig. 3.8, majority of the characters (such as "ABDEFIKLRSTUVW") have identical wrist

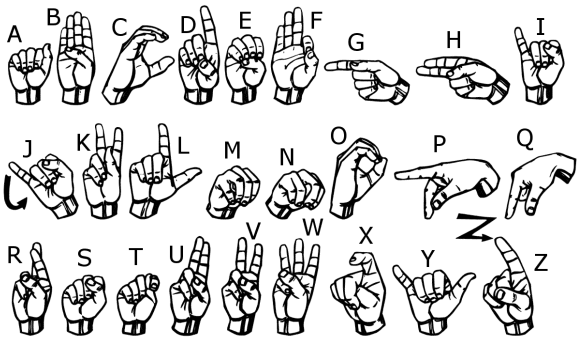


Figure 3.8: Hand poses for alphabets [2]

poses (positions and orientations). In cases where poses are different, the difference is too subtle to be reliably identified with variations in signing by different users. Thus, we realize that the watch data is not much useful. (2) Many characters have similar index finger poses ("GH","CO","BLX", "AISY"). Since the ring is only worn on the index finger, the ring data may not disambiguate among such characters. Nevertheless, ring data captures more information about the signed character than the smartwatch data. Hence, we use only the ring sensor data for finger-spelling detection.

Towards finger-spelling detection, we begin by collecting training data from the ring by a single user for all the 26 characters. No new training is needed for a new user. While the ring data does not sufficiently differentiate among characters, we exploit the English dictionary for detecting the whole words even though the individual characters may not be detected reliably in isolation.

When a new user signs an unknown word, we first segment the IMU data to obtain the sensor data for each individual character using techniques similar to the discussion in Section 3.5.1. Now, we consider all words in the English dictionary with exactly same number of characters as the unknown word signed by the user – we call this "search space list". Suppose the length of the word signed by the user is 6 and the first word on the "search space list" is "sensor". We measure similarity between the training data for each of the 6 characters - "s","e","n","s","o" and "r" with the data from the 1st, 2nd, 3rd, 4th, 5th and 6th segmented characters respectively from the user data for the unknown word. We compare the orientation between the training and testing data for measuring the similarity. Similarly, we derive the similarity metric with the second word on the "search space list", say "robust", and continue measuring similarity for all words in the dictionary. The dictionary word with the maximum similarity metric is decoded as the word signed by the user. We evaluate the idea with a dictionary size of 1000 words (Section 3.7). For working with larger dictionary sizes, the frequency of usage of the dictionary word may also be considered. If two dictionary words

have similar matching metrics, the one with higher frequency could be the more likely.

3.6 Putting it all together: System Architecture

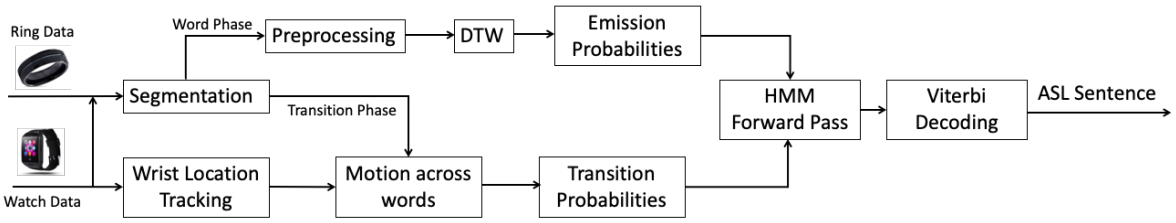


Figure 3.9: *FinGTrAC* Architecture

Fig. 3.9 depicts the overall architecture of *FinGTrAC*. The 9-axis IMU data from the ring and the smartwatch is the input. The input data is first segmented into "word-phases" and "transition phases". Preprocessing steps together with DTW is used to determine top 10 matches for data in the word-phase. The DTW metric is used to generate the emission probabilities for the HMM. The prior probabilities of word frequencies are also incorporated into the emission probabilities. The watch data is used to track hand motion between words in the context of a sentence. This generates the transition probabilities for HMM. The emission and transition probabilities form the input to HMM which then decodes the most likely sentence using the Viterbi algorithm. The decoded sentence is read on a microphone speaker or displayed on a phone screen.

3.7 Evaluation

User Study: All reported results in the paper are generated from a systematic user study campaign. Due to recruitment challenges, we could only recruit users with normal hearing ability. Thus, we conduct a study similar to guidelines followed in recently proposed wearable ASL translation works [119, 138]. We recruit users and train them extensively to perform ASL finger gestures through an online ASL tutorial. One user provides training data for 100 ASL words as discussed in Section 3.5 for computing the DTW similarity metric. We conduct the study with 10 test users that includes 7 males and 3 females. The volunteers are aged between 20-30, and weigh between 47kg to 96kgs. Overall, the users learnt 50 ASL sentences (3-11 words each). The sentences were composed from a dictionary of 100 most frequently used ASL words [114]. 53 of the words use both hands while 47 of the words use only the dominant

hand. The 50 sentences use a total of 90 words from this dictionary. Finally, the users perform gestures of the 50 sentences ten times by wearing the ring and smartwatch platform described in Section 3.3. We collect 9-dimensional IMU data both from the ring as well as the smart-watch during these experiments. The data is streamed to an edge device (desktop) which decodes sentences with a latency of 0.4 seconds.

Impact of Ring Position: Although the index finger is most involved in ASL gestures and

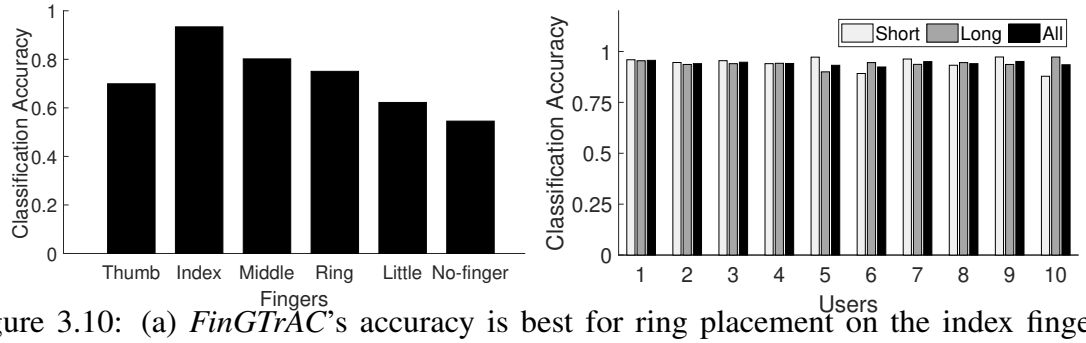


Figure 3.10: (a) *FinGTrAC*'s accuracy is best for ring placement on the index finger (b) Recognition accuracy is consistent across diverse users

thus we place the ring sensor on the index finger, we conduct a small scale study with just 3 users to verify how the position of the ring impacts the word classification accuracy. Fig. 3.10(a) shows the word classification accuracy as a function of position of the ring. The last-bar "No-finger" indicates that a ring sensor was not placed on any finger, and the classification was done only based on smartwatch data (for both DTW and HMM stages). Evidently, the index finger offers highest accuracy – upto 15% higher than the middle finger which offers second best accuracy. The accuracy is only 54% for the "No-finger" case. Thus, we conduct the study for the rest of the users by placing the ring sensor on their index finger. The next set of results presented has the ring sensor placed on the index finger.

Overall Sentence Recognition Accuracy: Fig. 3.11(a) shows the word error rate across different sentences. The *word error rate* of a sentence is the ratio of number of incorrectly decoded words and the total number of words. Most of the sentences are decoded correctly across users with an average word error rate of 5.86%. For cases where the words were not decoded correctly, Fig. 3.11(b) shows the histogram of the rank of the correct word. Evidently, the correct word is within top 5 most likely words as detected by our system. We believe this is a promising result which in combination with advances in NLP techniques can recover most of the residual errors.

Gesture Accuracy Breakup by Words: For each word gesture, the *word classification accuracy* is defined as the ratio of number of times the gesture was detected correctly to the total number of its occurrences. Fig. 3.11(c) shows a confusion matrix depicting the word

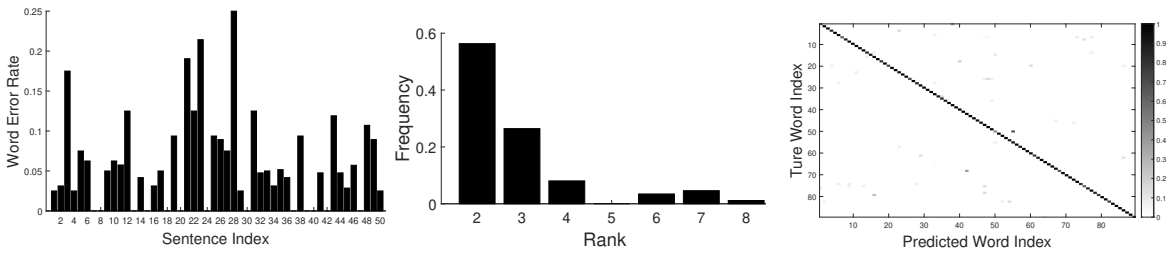


Figure 3.11: *FinGTrAC*'s Performance (a) Word error rate across sentences (b) Histogram of ranks of correct words (c) Confusion matrix for word classification – [1]

classification accuracy. Cases of miss-classifications ("cook" and "kitchen") happen when two words have very similar index finger motion and wrist location. In other miss-classification examples such as "water" and "fine", the gestures are similar enough that variations in user performance can affect the accuracy. However, most words are decoded correctly with an average accuracy of 94.2% .

One hand vs both hand gestures: Our dictionary includes 47 words that use only dominant hand and the other 53 words use both hands. We did not notice any difference in accuracy between these two classes. One handed gestures have a classification accuracy of 94.47% whereas two handed gestures have an accuracy of 93.85%.

Gesture Accuracy over Users: Fig. 3.10(b) shows the breakup of average word classification accuracy over users for short (4 words or less), long (5 words or more), and all sentences. Evidently, *FinGTrAC* is robust over diverse users. The results are also consistent for various sentence sizes.

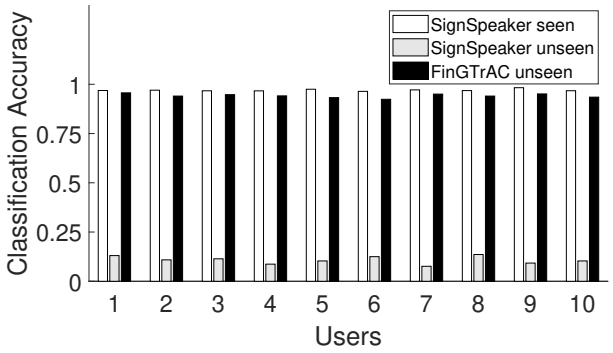


Figure 3.12: In contrast to SignSpeaker's deep learning approach, *FinGTrAC*'s HMM approach generalizes well to unseen sentences

Comparison with Related Work: We compare with the state of the art wearable ASL recognition in SignSpeaker [119] which detects 103 ASL gestures. Thus, our dictionary

sizes are similar for fair comparison. We implemented SignSpeaker's machine learning approach with Long Short Term Memory (LSTM) and Connectionist Temporal Classification (CTC).SignSpeaker trains the model with 11 users. The model is tested with 4 new users, however, the test users sign the same sentences signed by the other 11 users with no evaluation on new unseen sentences. We first reproduce this result by training the model with 9 users and testing with 1 user (10 fold cross validation [139]) for same sentences used in training. The accuracy is 97.1% which is roughly in agreement with results in SignSpeaker (Fig. 3.12). However, when we train the model with 49 sentences and test with a new unseen sentence (50 fold cross validation), we observe that the performance of SignSpeaker drops dramatically. While machine learning models are successful in speech recognition on unseen sentences, we believe the poor performance of machine learning based approach here is attributed to limited amount of training data in contrast to speech recognition models which have been trained with data accumulated over years. Our attempts to run the same machine learning algorithms on the ring data have also been unsuccessful in generalizing to unseen sentences for similar reasons. In comparison to SignSpeaker, FinGTrAC's accuracy for any unseen sentence is still 94.2% owing to its HMM based design that can detect any arbitrary composition of words from a given dictionary.

Gain by Module: Fig. 3.13(a) shows the improvements in accuracy delivered by various

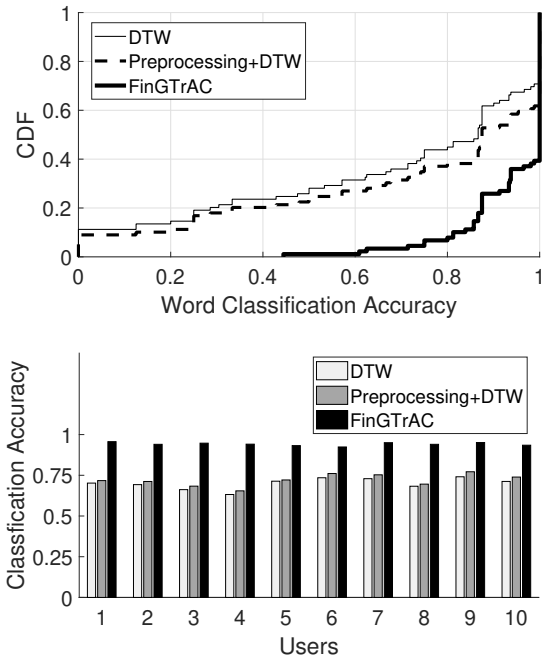


Figure 3.13: Classification accuracy improves with successive stages of optimization across for all users

modules in *FinGTrAC*. DTW detects words with an average accuracy of 70.1%. DTW is robust to accommodate variations when the overall gesture has similarities with the standard template. However, when the user performs the gesture completely differently, DTW can fail.The preprocessing stage looks at high level features and eliminates many of the wrong matches thus enhancing the accuracy to 72.2%. Although the average improvement with preprocessing looks marginal, the preprocessing steps can significantly improve the rank of correct words as discussed in Section 3.5.3. Further, the HMM model incorporates wrist location transitions in between words and improves the accuracy of word detection to 94.2% . Essentially, the application domain based inferencing with HMM has increased the gain fro 72.2% to 94.2% which we believe is significant. The improvement in accuracy by the technical modules is also consistent across various users (Fig. 3.13(b)).

	w1	w2	w3	w4
rank-1	restaurant	I	drink	month
rank-2	woman	have	month	milk
rank-3	three	my	sleep	game

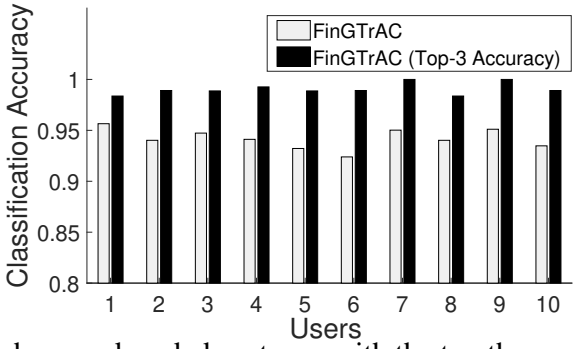


Figure 3.14: (a) Table shows a decoded sentence with the top three ranks for each word position. One of the rank-1 words is incorrect. *Can you guess the correct sentence?* Answer at end of paper[1](b) NLP techniques can boost the accuracy

Analysis of failure cases: *How bad is a failure case?* We note that most incorrectly decoded sentences have only 1-2 words that are incorrect. Even if the decoded words are incorrect, the correct word is mostly within the top 3 ranks after the HMM stage. An example is shown in the table in Fig. 3.14(a) where the top three ranking candidate words are shown for each of the decoded words in the sentence. Can you guess the correct sentence if we told you that only one word is wrong (i.e. correct word is not rank-1 but in top-3 ranks)? We show such examples to students in an informal setting – they were quick to identify correct sentences. Over the entire dataset, we observe that the correct word appears in top-3 ranks in 99.0% cases. As shown in Figure 3.14(b), if we assume that a word is classified correctly if it appears in the top-3 ranks, then the accuracy shoots up from 94.2% to 99.0%. We believe this is promising,

particularly because we observe that the incorrect sentences due to miss-classifications are often incorrect grammatically and semantically. However, if the correct word is not too far-away from the incorrect word in HMM's probabilistic ranking, we believe language models can be applied to further boost the accuracy by identifying semantically and grammatically correct sentences within the search space. This will be explored in future work (elaborated in Section 3.9).

Scalability Analysis: To evaluate the scalability of our system, we expand our user study by a modest amount. We augment the dictionary with 100 new words thus making the total size of our dictionary to be the top 200 popularly used ASL words [114]. The new 100 words were signed by one user using a procedure similar to the signing of the initial 100 words. This data is entered into the training database which now contains DTW templates for 200 popularly used ASL words. In addition to the 50 sentences from our initial dataset, we add 20 new sentences

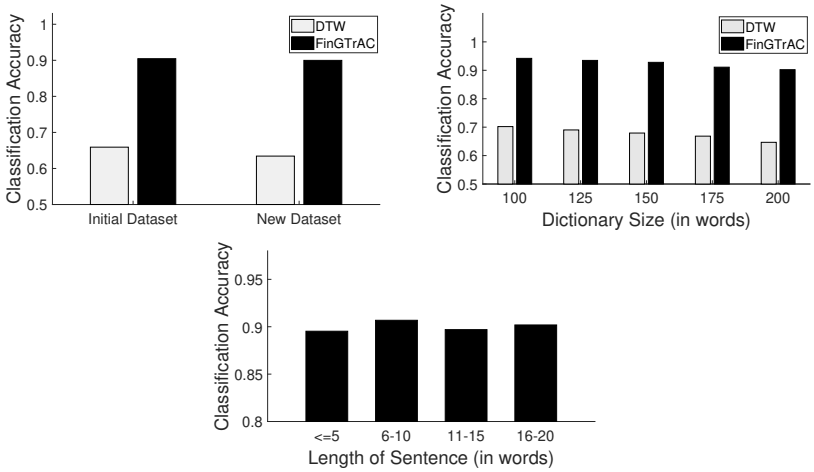


Figure 3.15: (a) Accuracy of newly signed sentences is comparable to sentences from the initial dataset (dictionary size = 200) (b) FinGTrAC's accuracy degrades gracefully (c) Accuracy is consistent irrespective of the length of the sentence

formed from 200 words in the dictionary. The word size over sentences range from 11-20 which is signed by 3 different users. We first validate whether the *word classification accuracy* of the initial dataset of 50 sentences is similar to the new dataset of 20 sentences in Fig. 3.15(a). Evidently, the accuracy levels are similar which indicates that FinGTrAC's performance is independent of the words used in forming sentences. However the accuracy is lower in both cases because of the expanded dictionary size of 200 words. Fig. 3.15(b) expands the scalability results of FinGTrAC with respect to the number of words in the dictionary. To ensure that all dictionary sizes contain all of the words used in all sentences, the results in Fig. 3.15(b) only uses the original 50 sentences (in the initial dataset) which were drawn from 100 most

popular words. However, we match them in a search space of varying dictionary sizes - 100, 125, 150, 175, and 200. A dictionary of size k contains k most frequently used ASL words. Fig. 3.15(b) shows the scalability results for both the DTW and the overall system FinGTrAC which integrates "DTW + HMM". Evidently, the *word classification accuracy* drops gracefully with more words added to the dictionary. While the DTW accuracy degrades from 70.1% to 64.7%, the overall accuracy of FinGTrAC degrades from 94.2% to 90.2%. Although the "DTW + HMM" approach itself may not be scalable for longer dictionary sizes, we believe language modeling approaches (Section 3.9) might help with further scalability. Fig. 3.15(c) shows the variation of accuracy with the length of the sentence (dictionary size = 200). We do not notice any trend in the graph because the "DTW + HMM" model in FinGTrAC is insensitive to the length of a sentence. Finally, the latency of processing varies between $0.4s$ to $1.1s$ as the length of sentences vary from 5 to 20 words, mainly because of the need to process a longer time series of data. However, we observe that the latency is insensitive to the dictionary size (over 100 to 200 words). This is because the HMM only looks at the top-10 ranked words (based on the DTW metric) regardless of the size of the dictionary (Section 3.5.3). On the other hand, the DTW matching stage, which is indeed affected by the size of the dictionary has negligible overhead in comparison to HMM.

Finger-spelling Detection: We consider a 1000-word dictionary of commonly used names (e.g. "John", "Adam"). We pick over 75 names from this dictionary and have users sign these names by ASL finger-spelling. We detect which of the 1000 words give the best match to determine the unknown word signed by a user. The names were decoded correctly with an overall accuracy of 96.6%. Fig. 3.16(a) shows the miss-classifications at the level of characters. Miss-classifications happen when two names differ in only one character (such as "Taylor" and "Naylor") and the distinguishing character has similar pose on the index finger. Fig. 3.16(b) shows the miss-classifications using a naive classifier that does not exploit the dictionary opportunities. By exploiting dictionary opportunities, *FinGTrAC* can reliably detect most non-ASL words.

3.8 Related Work

Gesture and Activity Recognition: IMU, WiFi, and Acoustic signals have been extensively used for gesture recognition in a number of applications [64, 136, 140, 141]. uWave [142] uses accelerometers for user authentication and interaction with a mobile device. Specifically, they use DTW based techniques to detect 9 gestures. FingerIO [143] uses acoustic reflections for user interface (UI) applications in small devices such as smartwatches. In particular, FingerIO

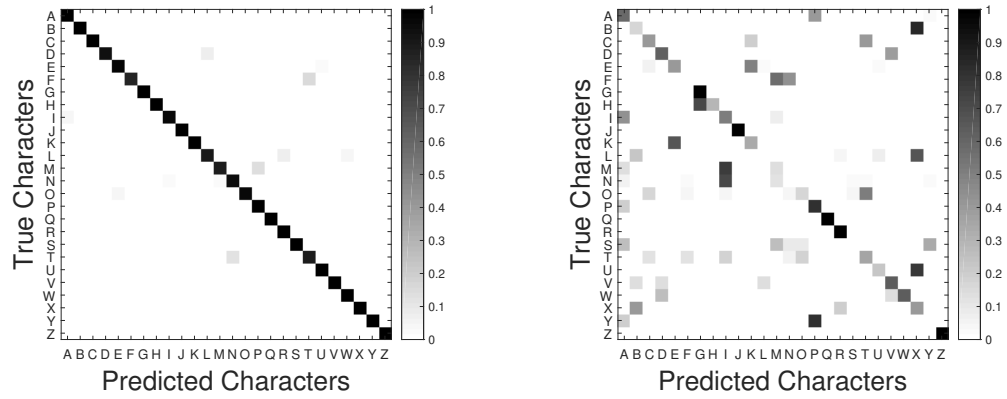


Figure 3.16: Confusion matrix of character classifications (a) With dictionary (b) Without dictionary

develops its technology based on phases of the reflected signals on orthogonal frequency-division multiplexing (OFDM) sub-carriers. FingerPing [112] uses acoustic signals for finger gesture detection. FingerPing places an acoustic-transducer and contact-microphone on the hand for transmitting acoustic signals through the body. They observe that the through-body channel varies with the finger gesture configuration using which they classify 22 different gestures. Capband [108] uses sensing with an array of capacitative sensors worn on the wrist. These sensors detect skin deformations due to finger motion thus being able to detect upto 15 hand gestures. RisQ [64] detects smoking gestures with wrist-worn IMUs using conditional random fields (CRFs). Work in [144] uses a fusion of 10 acoustic sensors (contact microphones) and IMU sensors for detecting 13 hand gestures in daily life. In contrast, *FinGTrAC* develops algorithms that are closely tied to the application for better accuracy with sparse sensors. Specifically while prior works with low intrusive sensors can only distinguish tens of gestures, FinGTrAC extends recognition to hundreds of gestures. Prior works specific to ASL are presented next.

Vision: Leap motion sensors, Kinect and computer vision research [145] can track fingers with cameras and depth imaging. Hat and neck-mounted cameras [138, 146] have been explored for ASL translation. Motion capture, depth imaging has been explored in [147, 148]. Deep learning approaches based on a hybrid of CNN and HMMs on video data is explored in [118]. 3D convolutional residual neural networks (3D- ResNets) are used for recognition of german and chinese sign languages from videos in [149]. More generally, recent papers [31, 32, 150] in computer vision can track 3D finger motion of users from videos using a combination of techniques in generative adversarial networks (GAN), convolutional neural networks (CNN), and weakly supervised learning techniques. While sophisticated computer vision techniques can easily track fingers and thus enable translation of larger ASL dictionary sizes, cameras do

not provide ubiquitous tracking and need clear view and lighting conditions. In contrast, our goal is to explore the limits and feasibility of sensing with wearable devices.

WiFi: WiFi based hand/finger gesture detection has been explored [45–47] that use wireless channel and doppler shift measurements for ASL recognition. WiFinger [45] uses channel state information (CSI) of wireless reflections with discrete wavelet transforms (DWT), and DTW for classifying upto 9 finger gestures. Work in [46] uses directional antennas and collects WiFi signal strength measurements of reflections from the body. Using pattern matching techniques, the work shows the feasibility of detecting 25 finger gestures. Work in [47] uses WiFI CSI together with feature extraction and SVM techniques to distinguish upto 5 gestures. Wisee [51] uses doppler shifts of wireless reflections to classify 9 different gestures. SignFi [54] is an innovative work that uses wireless channel measurements from WiFi APs for ASL recognition over a larger dictionary. We admit that WiFi based detection is completely passive without on-body sensors. However, we believe wearable approaches can complement WiFi based techniques in an ubiquitous and adhoc setting (ex: outdoors) with no availability of WiFi.

Wearable Sensors including Gloves: An extensive review of glove systems is provided in [15]. The survey indicates that works between 2007 to 2017 use a combination of one handed and two handed gloves with accelerometers, flex sensors and EMG sensors to detect upto 100 ASL words. Flex sensor technology [151, 152] is popularly used for detecting finger bends. These sensors are typically made of resistive carbon elements whose resistance depends upon the bend radius. Works [153–155] use flex sensors on each of the fingers to be able to distinguish upto 40 sign language gestures. Optical resistance based finger bend technologies have also been adopted [156, 157]. By combining a light emitting diode with a light dependent resistance, the optical technology detects the finger bend by observing the variation in resistance due to variation in light intensity resulting from finger bending. Work in [158] uses 5 optical flex sensors for detecting 25 words in Malaysian sign language using probabilistic classification models. Similarly work [159] uses a glove made of optical flex sensors in detecting 25 words and basic postures in Korean Sign Language. Tactile sensor technology [160] detects whether a finger is curled or straight by using a robust polymer thick film device. The resistance of the material decreases as pressure on the finger increases thus sensing the state of the finger. Work in [154] detects upto 8 ASL gestures using such tactile sensors. Hall Effect Magnetic Sensors [161] are used for ASL recognition over 9 ASL words using logistic regression algorithms. The core idea is to place a strong magnet on the palm, and place small magnetic sensors on finger tips. As the finger bends, it comes closer to the palm and hence closer to the strong magnetic field. By sensing this, the bending of fingers can be detected for use in gesture classification. Works with Electromyography (EMG) sensors [162, 163]

typically use tens of electrodes to detect muscle activity for ASL word recognition using support vector machines, nearest neighbor, naive bayes, and other classification algorithms. In addition, a number of glove based systems which combine the above sensors with IMUs have been proposed with some of them being commercially available on the market. 5DT Data Glove uses [164] flex sensors on all fingers to measure bending angles of fingers. Using this, works [165] have shown detection of upto 26 ASL alphabets. Work in [166] uses commercially available CyberGlove [37] as well as IMU sensors to detect ASL words from a dictionary of 107 words. In contrast to above works that use 5-20 of embedded sensors on hands, *FinGTrAC* uses a low intrusive platform of smartwatches and rings to detect ASL gestures with a similar accuracy without compromising on the size of the dictionary. Closest to our work is a recently proposed innovative system called SignSpeaker [119] that detects sentences composed of 103 ASL words using smartwatch. While SignSpeaker is less intrusive than *FinGTrAC* given it only requires a smartwatch, we believe *FinGTrAC* differs from SignSpeaker in an important way. FinGTrAC can detect unseen sentences whereas SignSpeaker cannot detect unseen sentences (Evaluated in Section 3.7) with the evaluation being only done with seen sentences. Thus, we believe the non-deep learning approach in FinGTrAC generalizes well in comparison to SignSpeaker where training needs to be done for all possible sentences which entails exponentially larger training overhead in the size of the dictionary. While deep learning with RNNs/LSTMs [120] is popular in speech recognition, the success is attributed to large training datasets accumulated over years. We discuss potential methods to integrate deep learning into our problem domain in Section 3.9. Work [167] uses 6 rings for ASL translation but lacks in any detail. AuraRing [168], a recent work, tracks the index finger precisely using a magnetic wristband and ring. We believe *FinGTrAC* can integrate such new hardware innovations into the model for improving the accuracy in the DTW stage.

3.9 Discussion and Future Work

Implement-ability in Real Time: *FinGTrAC* involves three computation modules. (1) DTW for word classification/ranking (2) HMM and viterbi decoder for sentence decoding (3) Wrist location tracking. The first two only involves low weight computation. Unlike wireless communications where the packet sizes are long, the size of a sentence is only a few words making viterbi algorithm tractable. The third module is perhaps the most complex one since it involves particle filters. Currently, we can run the particle filter on a desktop machine/edge device in real time. Given our bandwidth requirement is minimal 50kb/s, FinGTrAC operates in real time with a sentence decoding latency of 0.4 seconds, and a battery energy consumption

of 150mW with WiFi streaming [169].

Other IoT Applications: Domain specific modeling of ASL helps improve the accuracy of gesture tracking. Similarly, we will also consider finger motion tracking for other applications by exploiting application-specific context. Analysis of finger motion data of tennis, baseball, basketball players can provide valuable information for coaching and injury management, as well as provides bio-markers of motor diseases [170].

Limitations and Future Work: We believe this is a significant first step, yet full ASL translation is a complex problem. In particular, the small dictionary size is a limitation since 200 ASL words do not offer sufficient coverage for practical usage in real world. There are also other limitations including not handling facial expressions and inability to exploit the rich ASL grammar. Therefore, we will explore the following approaches to address these limitations in the future.

(1) Non deep learning techniques such as Natural Language Processing (NLP) and Semantic Context to augment dictionary: FinGTrAC benefits from NLP. For example, a sentence *"You can hear me?"* was wrongly decoded as a grammatically incorrect sentence: *"You can more me?"* In fact, the miss-classified word ("hear") was the second most likely word. Thus, we believe that a deeper integration of NLP techniques can improve accuracy to substantially expand the dictionary size of *FinGTrAC*. Language modeling techniques [171] from speech recognition will also be considered. The semantic context of a sentence offers rich information to correct miss-classifications. A sentence - *"Sleep my bed don't_mind"* was classified as *"Think my bed don't_mind"*. Clearly, the first sentence is more meaningful than the second. We plan to incorporate context information not only within sentences but also across sentences using similarity metrics from word2vec [172], WordNet [173], context from FrameNet [174] etc. We will first attempt to directly incorporate such NLP constraints not only within sentences but also across sentences to augment the dictionary size as much as possible. However, as an alternative we will also explore deep learning approaches elaborated next.

(2) Deep Learning to Enhance Dictionary: Unlike speech processing or image recognition, no large dataset is available for tracking finger motion with IMU. Thus, FinGTrAC uses HMM to minimize training overhead and validate the feasibility. RNNs [171] are very popular in speech processing and language translations. With enough training data accumulated over years, they can learn complex functions as well as intelligently use old observations in decision making using LSTMs [120]. To bootstrap training data generation, we will exploit finger motion from videos [31] of ASL tutorials and news. We believe this will substantially increase the size of *FinGTrAC*'s dictionary.

(3) Facial Expressions: Main expressions of interest include lowering/raising eyebrows;

anger, happiness, sadness; jaw-motion; winking; nodding; head-tilt etc. We will explore ear-worn sensors (IMU and biological sensors) to track such expressions [175, 176]. Ear-worn sensors can be used naturally as earphones or ornaments – the design would be non-intrusive. EMG (facial muscles), EEG (brain signals) and EOG (eye signals) can be integrated easily into earphones [177] which provide opportunity for sensing facial expressions. While prior works detect isolated expressions, we plan to integrate the sensing deeply into language and machine learning in the future study.

(4) User study with expert ASL users: Lack of testing with fluent ASL users is a limitation of our system. However, we believe that the FinGTrAC's results are promising enough to develop on our ideas above for enhancing dictionary size and incorporating facial expressions such that the system is feasible for full fledged ASL translation. We plan to validate such an end-to-end ASL translation system with expert ASL users.

(5) Finger Motion Tracking and Additional Sensors: Towards pushing the limits of accuracy, *FinGTrAC* will explore free form tracking of finger joint locations. While this is non-trivial with under-constrained information, the anatomical constraints of fingers open up opportunities [5]. Modeling the constraints dramatically decreases the search space for finger poses thus making it feasible to track. We will also measure the trade-offs in intrusiveness and accuracy in using additional sensors or a second ring. We also consider through body acoustic vibrations from ring to smartwatch [112] or integrating a single EMG electrode into wrist-watch for 3D finger motion tracking while still keeping the sensor footprint small.

3.10 Conclusion

This paper shows the feasibility of finger gesture classification by using low-weight wearable sensors such as a smart-ring and smart-watch which are becoming popular in recent times. For such platforms, the importance of application context in scaling gesture recognition from tens to hundreds of gestures has been demonstrated through a case-study with ASL translation. Previous approaches which use motion tracking cameras cannot offer ubiquitous tracking, while wearable solutions that require sensor-gloves, or EMG sensors are too cumbersome. In contrast, *FinGTrAC* shows feasibility with a ubiquitous and easy-to-use platform with minimal training. In building the system, *FinGTrAC* uses a probabilistic framework incorporating the noisy and under-constrained motion sensor data, as well as contextual information between ASL words to decode the most likely sentence. A systematic user study with 10 users shows a word recognition accuracy of 94.2% over a dictionary of 100 most frequently used ASL words. Despite progress, this is only the first step. We believe the results from this work offer

significant promise in extending this work. Opportunities in sensing, machine learning, and natural language processing can be exploited to extend this work towards enabling a realistic solution with low intrusive wearables for seamless ASL translation.

Chapter 4
NeuroPose

4.1 Introduction

3D finger pose tracking enables a number of exciting applications in sports analytics [13], healthcare and rehabilitation [14], sign languages [15], augmented reality (AR), virtual reality (VR), haptics [16] etc. Analysis of finger motion of aspiring players can be compared to experts to provide automated coaching support. Finger motion stability patterns are known to be bio-markers for predicting motor neuron diseases [17]. AR/VR gaming as well as precise control of robotic prosthetic devices are some of the other applications that benefit from 3D finger pose tracking [20, 21].

Web-based augmented/virtual reality applications are becoming popular [178, 179] leading to standardizations of WebXR APIs [180]. Examples include remote surgery, virtual teaching (body-anatomy, sports, cooking etc), multiplayer VR gaming. These applications involve augmenting the context of the user (location, finger-pointing direction etc.) with information from the web (on-screen-viewport, textual-information, haptic stimulation etc.). Finger motion tracking is a common denominator of such applications.

Motivated by the above applications, there is a surge in recent works [31, 32] in computer vision that track 3D finger poses from monocular videos. Given they do not require depth cameras, the range of applications enabled is wide. However, vision based techniques are affected by issues such as occlusions and the need for good lighting conditions to capture intricate finger motions.

In contrast to vision, the main advantage of wearables is in enabling ubiquitous tracking without external infrastructure while being robust to lighting and occlusions. While data gloves [37–39] with IMU, flex, and capacitative sensors have been popularly used for finger motion tracking, it is shown that gloves hinder with dexterous hand movement [40]. As alternatives to putting sensors on fingers, sensing at wrist with surface acoustic [112], capacitative [108],

bioimpedance [181], ultrasonography [182], wrist pressure [183] etc., has been explored, but the sensing is only limited to tens of gestures. Beyond discrete gestures, infrared [184] and thermal cameras [33] mounted on wrist have been explored for continuous 3D pose tracking, but has limitations on hand motion. In contrast, we explore using ElectroMyoGraphy (EMG) sensors worn like a band on the forearm (Fig. 4.5) with the following advantages: (i) Captures information directly from muscles that activate finger motions, thus offering rich opportunities for continuous 3D finger pose sensing (ii) A user does not need to put sensors on fingers and thus she is able to perform activities requiring fine precision (iii) Tracking is independent of ambient conditions of lighting or presence of objects in the background. (iv) EMG sensors can measure emotions (like fear) and muscle strain to make VR tasks on safety (fire, construction etc) and physical-activities (e.g. rock climbing) more realistic [185, 186].

Despite the benefits, EMG sensors are not as popular as smartphones or smartwatches. Thus the user needs to carry a separate EMG band with her. Nevertheless, we believe there are motivating applications (prosthetic devices for amputees, sports coaching, augmented reality) where a user can selectively wear the device when needed instead of constantly wearing it. The prospects of adoption of EMG sensing for AR/VR is on the rise (led by Facebook [187, 188]) because EMG can pick strong/unambiguous signals of minute finger motion. Thus, we believe understanding the limits and bounds of sensing can help develop interesting applications and use-cases encouraging better social adoption.

Prior works on EMG based finger motion tracking are limited to tracking a few hand gestures [71–75], or tracking hand poses over a set of discrete gesture related motions [74, 75]. They do not provide free form 3D pose tracking for arbitrary hand motion. This paper proposes a system called *NeuroPose* that fills this gap in literature by designing a EMG wearable-based 3D finger pose tracking technology. Towards this end, *NeuroPose* uses an off the shelf armband consisting of 8 EMG channels (Fig. 4.5) for capturing finger motion and converting it into 3D hand pose as depicted in Fig. 4.1. Using only two of the eight channels might increase the comfort of wearing the sensor with a modest loss in accuracy.

Briefly, the EMG sensors capture neural signals propagating through the arms due to finger muscle activations. Each finger muscle activation generates a train of neuron impulses, which are the fundamental signals captured by the sensors. Given such EMG sensor measurements, tracking the 3D pose is non-trivial and introduces a number of challenges: (i) Human hand is highly articulated with upto 21 degrees of freedom from various joints. The complexity of this search space is comparable to tracking joints in the skeletal model of a human body. (ii) Impulses from multiple fingers are mixed in complex non-linear patterns making it harder to decouple the effect of individual fingers from the generated sensor data. (iii) The strength of the

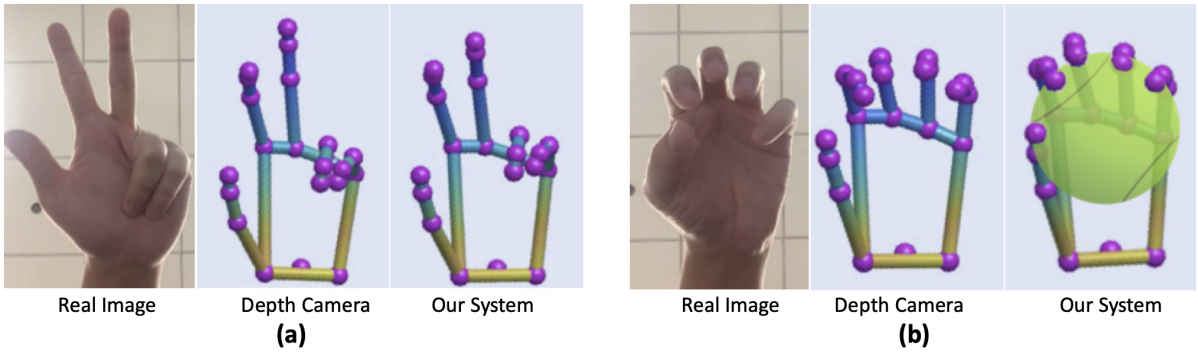


Figure 4.1: A comparison between a real image, a depth camera, and *NeuroPose*. Tracking of fine grained hand poses can enable applications like: (a) Word recognition in sign languages (b) Augmented reality by enhancing the tracking output. A short demo is here [3].

captured signals depends on the speed of motion, and finger pose. (iv) The nature of captured data varies across users due to variations in body sizes, anatomy etc. (v) The sensor data is noisy due to hardware imperfections.

In handling the above challenges, *NeuroPose* exploits a number of opportunities. (i) Finger motion patterns are not random but they follow tight anatomical constraints. Fusion of such constraints with the actual sensor data dramatically reduces the search space. (ii) Innovation in machine learning (ML) algorithms that explicitly and implicitly fuse such constraints with sensor data have been exploited. In particular, *NeuroPose* explores architectures in Recurrent Neural Networks (RNN) [171], Encoder-Decoder [189], ResNets [190] in achieving a high accuracy. (iii) A transfer learning framework based on *adaptive batch normalization* is exploited to learn user dependent features with minimal overhead for adapting a pretrained model to a new user for 3D pose tracking.

NeuroPose is implemented on a smartphone and runs with a latency of $0.1s$, with low power consumption. A systematic study with 12 users achieves an accuracy of 6.24° in median error and 18.33° in the 90%-ile case. The accuracy is robust to natural variation in sensor mounting positions as well as changes in wrist positions of users.

Our contributions are summarized below:

(1) NeuroPose shows the feasibility of fine grained 3D tracking of 21 finger joint angles using EMG devices for arbitrary finger motions.

(2) Fusion of anatomical constraints with sensor data into machine learning algorithms for higher accuracy.

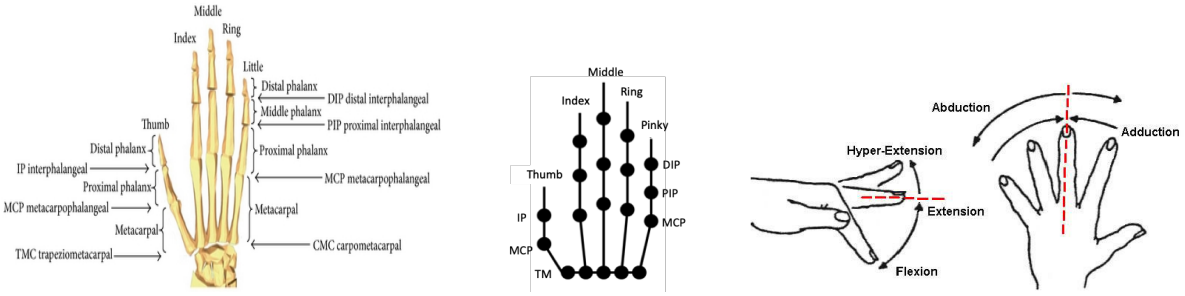
(3) Implementation on embedded platforms and extensive evaluation over diverse users.

4.2 Background

We begin with a brief overview of: (i) the anatomical model of the human hand (ii) the neuro-muscular interactions during finger muscle activations and how it manifests as EMG sensor data.

4.2.1 Hand Skeletal Model

The human hand consists of four fingers and a thumb which together exhibit a high degree of articulation. Fig.4.2(a) depicts the skeletal structure of the hand with various joints that are responsible for complex articulation patterns that generate 3D hand poses. Fig. 4.2(b) shows a simplified kinematic view. The four fingers consist of MCP (metacarpophalangeal), PIP



((a)) ((b)) ((c))

Figure 4.2: (a) Anatomical details of the hand skeleton [4] (b) Kinematic structure and joint notations [5] (c) Finger motions include flex/extensions and abduction/adductions [6]

(proximal interphalangeal), and DIP (distal interphalangeal) joints. The joint angles at PIP (θ_{pip}) and DIP (θ_{dip}) joints exhibit a single degree of freedom (DoF) and can flex or extend (Fig.4.2(c)) the fingers towards or away from the palm.

In addition to flexing, the MCP joint can also undergo adduction and abduction (side-way motions depicted in Fig.4.2(c)), and thus possesses two DoFs, denoted by $\theta_{mcp,f/e}$, and $\theta_{mcp,aa}$ respectively. Thus, each of the four fingers posses four DoF. The thumb on the other hand exhibits a slightly different anatomical structure in comparison to the other four fingers. The IP (interphalangeal) joint can flex or extend with a single DoF (θ_{ip}). The MCP and TM (trapeziometacarpal) joints possesses both flex and abduction/adduction DoF, thus the thumb has five DoF – θ_{ip}, $\theta_{mcp,f/e}$, $\theta_{mcp,aa}$, $\theta_{tm,f/e}$, and $\theta_{tm,aa}$. The other 6 DoF comes from the motion of palm including translation and rotation. We ignore the motion of the palm in this paper and only focus on tracking fingers which together have 21 DoF – modeled as 21 dimensional space ($\mathbb{R}^{21}$). Thus, *NeuroPose*'s goal is to track this $\mathbb{R}^{21}$ dimensional space to capture the 3D finger pose.

The various joint angles responsible for finger articulation exhibit a high degree of correlation and interdependence [4, 5]. Some of the intra-finger constraints are enumerated below:

$$\theta_{dip} = \frac{2}{3}\theta_{pip} \tag{4.1}$$

$$\theta_{ip} = \frac{1}{2}\theta_{mcp,f/e} \tag{4.2}$$

$$\theta_{mcp,f/e} = k\theta_{pip}, \quad 0 \leq k \leq \frac{1}{2} \tag{4.3}$$

Equation 4.1 suggests that in order to bend the DIP joint, the PIP joint must also bend under normal finger motion (assuming no external force is applied on the fingers). Likewise, Equation 4.2 is a constraint on the thumb joints. Similarly, the range of motion for PIP is very much limited by the MCP joint (Equation 4.3). The generic range of motion constraints for other fingers are enumerated below:

$$\begin{aligned} -15^\circ \leq \theta_{mcp,aa} \leq 15^\circ \\ 0^\circ \leq \theta_{dip} \leq 90^\circ \\ 0^\circ \leq \theta_{pip} \leq 110^\circ \end{aligned} \tag{4.4}$$

Clearly, abduction/adduction angles have a smaller range of motion compared to flex/extensions. In addition to these constraints, there are complex inter-dependencies between finger joint motion patterns which cannot be captured by well formed equations. However, our ML models will be able to automatically learn such constraints from data and exploit them for high accuracy tracking.

4.2.2 Electromyography Sensor Model

Electromyography sensors can detect electrical potential generated by skeletal muscles due to neurological activation. Such signals can provide information regarding temporal patterns and morphological behaviour of motor units that are active during muscular motion [191]. Not only are the signals useful for detecting and predicting body motion induced by the muscles but also useful for diagnosis of various neuromuscular disorders and understanding of healthy, aging, or fatiguing neuromuscular systems.

Muscles of Interest: We now provide a brief overview of muscular involvement during

finger motions. Several muscles are involved in performing finger motions. Fig. 4.3(a) and (b) depict the anatomical structure of the human arm. *Extensor Pollicis Longus* extends the thumb joints whereas *Abductor Pollicis Longus and Brevis* performs thumb abductions. *Extensor Indicis Proprius* extends the index finger. *Extensor Digitorum* extends the four medial fingers and *Extensor Digiti Minimi* extends the little finger. *Volar interossei* and *Dorsal interossei* group of muscles are responsible for adduction and abduction respectively of index, ring, and little fingers towards/away from the middle finger. They are connected to *proximal phalanx* and the *Extensor digitorum*. *NeuroPose* mainly focuses on such muscles that perform finger actions. Other muscles that are involved in large scale motion and supporting strength include *Supinator* for forearm motion, *Anconeus and Brachioradiali* for elbow joint, *Extensor Carpi Ulnaris*, *Extensor Carpi Radialis Longus and Brevis* for wrist joint etc.

Feasibility of Tracking the Muscles of Interest: Among the targeted muscles of interest, although some of them appear close to the skin surface, some of them are deep (such as Extensor Indicis). Therefore, a natural question to ask is: *Is surface EMG alone sufficient to capture all such muscles of interest?* To verify this, we conduct a simple experiment where we flex and extend each of the five fingers, and observe the activity on the EMG channels. Depicted in Fig. 4.4, all fingers show noticeable activity on the EMG channels for flex/extensions (the activity on channel number 1 is shown per conventions in Fig. 4.5.) For sake of brevity, we provide one example for abduction/adduction in Fig. 4.4(f) for abducting/adducting all fingers together however, we note that each finger individually generates a noticeable pattern for abduction/adduction motions. An important observation from the figures is that the muscle group responsible for motion of index finger – *Extensor Indicis*, a non-surface muscle group relative to sensor placement in Fig. 4.5 – also generates a noticeable spike in the EMG channel data (Fig. 4.4(b)). This is also validated by prior research related to deep muscle activity [192]. These signals must be carefully analyzed further to capture the precise magnitude of finger joint angles, particularly when multiple fingers are simultaneously in motion. Towards the end, we begin by describing the interference pattern on the EMG sensors by signals from different muscle groups. Separating out the individual finger motions from such EMG sensor data will be discussed in Section 4.4.

Biological Model: We now provide a brief description of the biological model of EMG signals generated due to muscle activations (illustration in Fig. 4.3(c)). Muscles consist of fundamental units called muscle fibres (MF) which are the primary components responsible for contraction. Activation of an MF by the brain results in propagation of an electrical potential called action potential (AP) along the MF. This is called motor fibre activation potential (MFAP). The MFs are not excited individually but are activated together in groups called motor units.

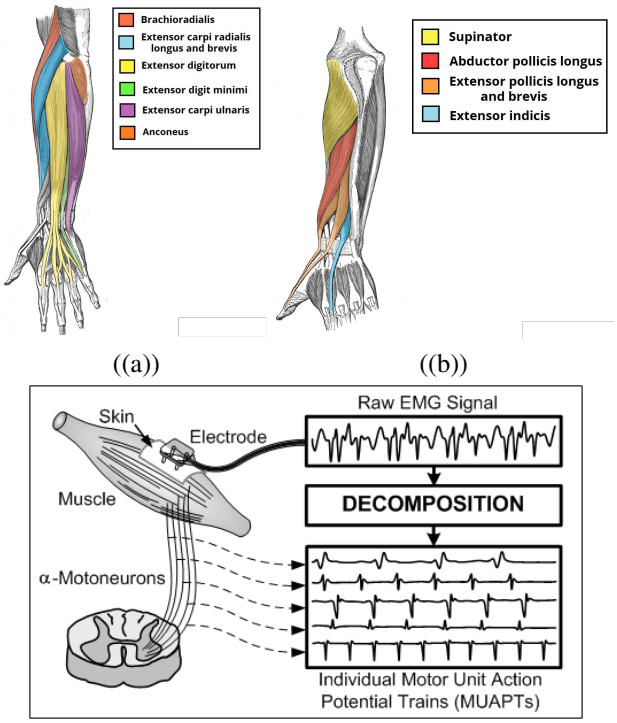


Figure 4.3: (a) and (b) Anatomical details of forearm muscles [7] (c) EMG signals from an electrode can be decomposed into constituent motor unit action potential trains (MUAPT) [8]

Groups of motor units coordinate together to contract a single muscle. Individual MFAPs cannot be detected separately, instead summation of all MFAPs within the motor unit generates a signal called as motor unit action potential (MUAP) as shown in the below equation

$$MUAP_j(t) = \sum_{i=1}^{N_j} MFAP_i(t - \tau_i)s_i, \tag{4.5}$$

where τ_i is the temporal offset, N_j is the number of fibres in motor unit j, and s_i is a binary variable indicating whether or not the muscle fibre is active. The temporal offset depends on the location of the muscle fibre. The number of observed MFAPs within a MUAP also depends on location of EMG electrode because the potential generated by far away fibres are typically detected in attenuated form at the electrode. A similar muscle action can result in different shape of the generated MUAP signal depending on the previous state of the muscle as well as the temporal offset τ_i which can vary.

The above equation represents a single instance of firing, but the motor units must fire repeatedly to maintain the state of muscle activation. Continuous muscle activations can generate a train of MUAP impulses separated by inter discharge intervals (IDI), as depicted in

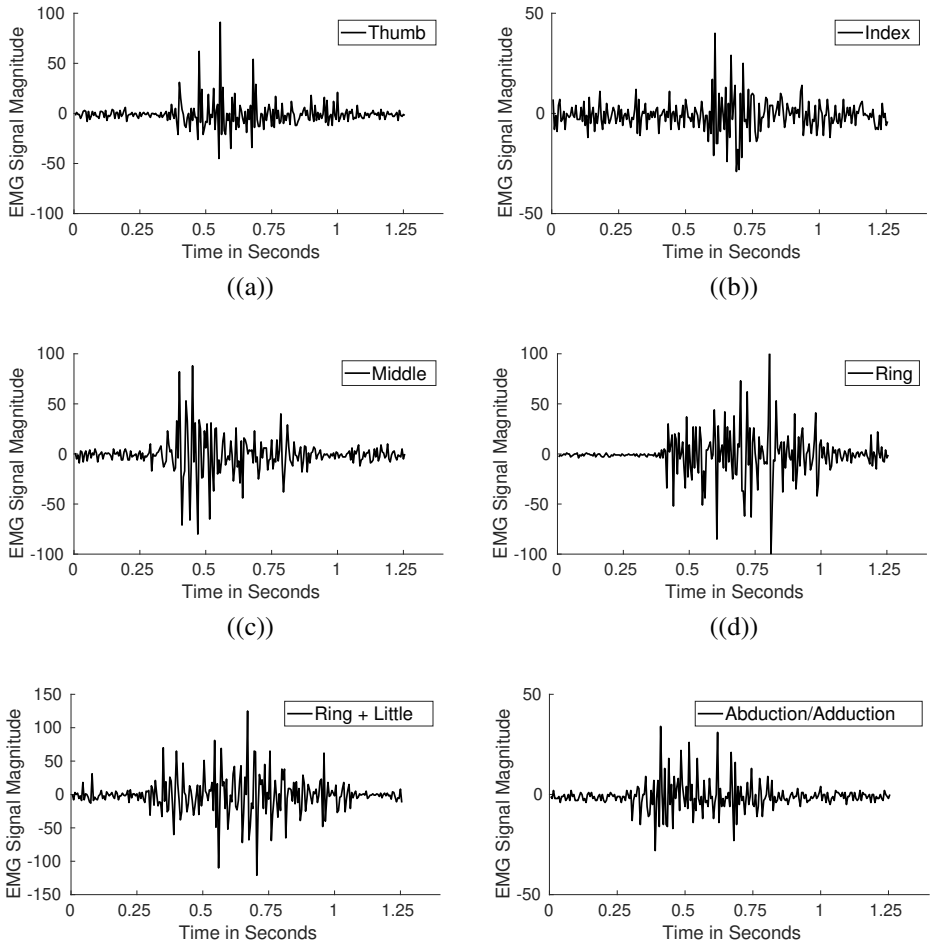


Figure 4.4: Flex/extension motion of all fingers generate noticeable "spike" in the EMG data for (a) Thumb (b) Index (c) Middle (d) Ring (e) Little + Ring (the little finger cannot be flexed without jointly moving the ring finger) (f) Abduction/Adduction of all fingers

the below equation

$$MUAPT_j(t) = \sum_{k=1}^{M_j} MUAP_{jk}(t - \delta_{jk}), \tag{4.6}$$

where M_j is the number of times the jth motor unit fires, δ_{jk} is the kth firing time of the jth motor unit.

Finally, the electric potential detected at an EMG electrode is the superimposition of signals by spatially separated motor units and their temporal firings patterns dependent on their respective IDIs. This spatio-temporal superimposition is depicted in the below equation where $n(t)$ is the noise term, and N_m is the number of active motor units.

$$EMG(t) = \sum_{k=1}^{N_m} MUAPT_j(t) + n(t). \tag{4.7}$$

While in theory, the EMG signal is composed of activation from every single muscle fibre, in practice the electrode can only detect the signals from fibres closer to the electrode because the signals attenuate below noise level with distance. Our EMG sensor platform described next exploits multiple electrodes to capture activations of all fibres involved in finger motion. Once the EMG data is captured, the core technical challenge is in decomposing the signals into activations responsible for individual joint movements. Towards this, we introduce ML algorithms in Section 4.4 for signal decomposition.

4.3 Platform Description

Figure 4.5: (a) 8 channel Myo armband (b) Myo armband in action

Our platform includes a MYO armband depicted in Fig. 4.5 worn on the arm. It consists of 8 EMG channels, as well as Inertial Measurement Unit (IMU) sensors of accelerometers, gyroscopes, and magnetometers. The data is streamed wirelessly over bluetooth to a desktop/smartphone device. *NeuroPose* is implemenetd on a sony xperia z3 dual smartphone that captures the EMG data and provides finger motion tracking results. The MYO sensor is low-cost, and appears to be solidly built. Although the MYO armband fits perfectly aesthetically on the arm it might seem intrusive for some users. Towards minimizing the intrusiveness of the platform, *NeuroPose*'s implementation with only a 2-channel EMG data offers a low-intrusive option with a modest loss in accuracy (Section 4.5).

Skin Temperature Calibration: The EMG amplitude may be slightly affected by skin temperature variations [193]. The surface Myo platform warm ups the contacted muscle [194] slightly. This helps the sensor to form a stronger electrical connection with the muscles to minimize the effects of temperature.

Other Platforms: We note that unlike smartwatches or smartphones, there is no globally

acceptable platform for EMG sensing yet. Facebook has recently acquired patents related to MYO armband [187, 188] for developing finger tracking technology for its thrust towards AR/VR applications. Other form factors ranging from arm-bands, tattoos, and arm-gloves have been proposed by both academia and industry with no consensus on what is best [71, 72, 80, 195, 196]. Therefore, the ML models developed in this paper may not apply directly to a hardware of different form factor than what is used here. While there are uncertainities about what platforms will gain wide spread adoption, our goal is to show that enough information exists in surface EMG data for continuous tracking of arbitrary finger motions. Furthermore, by showing the right applications and use-cases, we believe we can influence the process of convergence of hardware platforms.

4.4 Core Technical Modules

We explore multiple ML models for 3D finger motion as elaborated in this section.

4.4.1 Encoder Decoder Architecture

In order to generate plausible finger pose sequences with spatial constraints across fingers, as well as temporally smooth variations over time, we design an encoder-decoder network as illustrated in Fig. 4.6. Specifically, the network captures a holistic view of a large interval

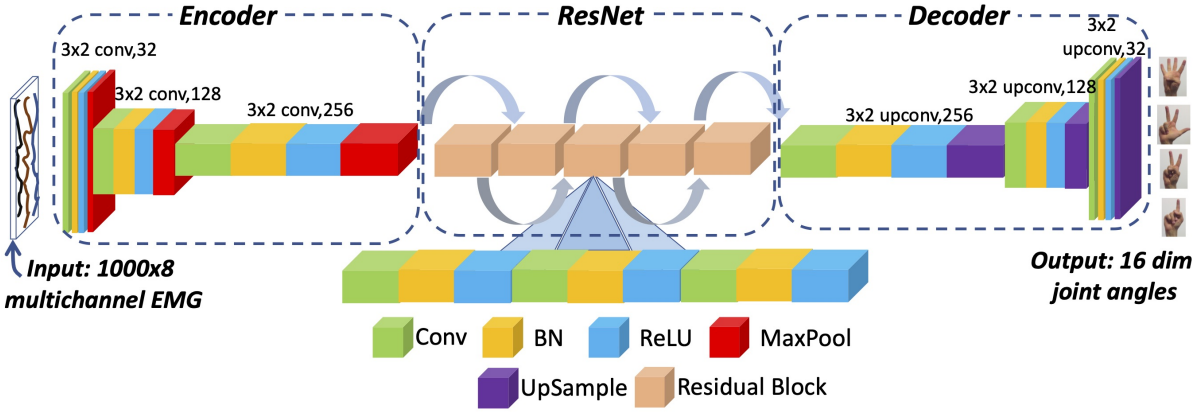


Figure 4.6: Encoder Decoder Architecture used in *NeuroPose*:

of time-series sensor data instead of a single sensor sample. This enables the network to enforce and learn the key spatio-temporal constraints as well as consider historical EMG data while making hand pose inferences. The network accepts $5s$ of sensor data and outputs

the corresponding 3D hand pose sequence. The various components of the architecture are elaborated next.

Encoder: The encoder-decoder model maps a sequence of input EMG data to a sequence of 3D finger poses. Unlike discrete classes, the output space of the model is a continuous domain $\mathbb{R}^{21}$. Among these 21 dimensions, 5 of the dimensions (θ_{dip} for four fingers and θ_{ip} for thumb) can be directly computed using Equations 4.1, 4.2. Thus, the actual output of the network is only 16 dimensions – $\mathbb{R}^{16}$.

While one possibility is to build a network with a series of convolutional layers, this will increase the number of parameters in the network, thus causing issues not only in compute complexity and memory but also in convergence. Thus, the encoder uses a series of downsampled convolutional filters. This captures a compact representation of the input which will later be used by the decoder in generating 3D hand poses.

The input x to the encoder is a multi-channel EMG data of dimensions $T \times 8$, where we choose $T = 1000$, which at a sampling rate of 200Hz translates to a duration of 5s. The encoder consists of a series of CONV-BN-RELU-MAXPOOL layers, which are elaborated below: (i) The *CONV* sub-layer includes 2D convolutional filters that perform a basic convolution operation [197]. The *CONV* sub-layer extracts spatio-temporal patterns within EMG data to learn features representative of finger motions. (ii) This is followed by a batch normalization (*BN*) sub-layer whose role is to accelerate convergence of the model by controlling huge variations in the distribution of input that passes from one layer to the next [198]. (iii) The *BN* module is followed by an *activation* sub-layer, which applies an activation function to the output of the *BN* layer. We chose a Rectified Linear Unit *(ReLU)* activation function [199]. While non-linearities are critical in training a deep neural network, among possible alternatives ReLU is popular because of its strong biological motivation, practicality of implementation, scale in-variance, better gradient propagation etc. We also add dropouts [200] following RELU activations. They serve as an adaptive form of regularization which knocks off some of the parameters of the network with a random probability of 0.05. (iv) Finally, max-pooling is applied to the output so as to downsample the feature size toward reaching a compact feature representation of the EMG data. Max pooling is done by applying a max filter to non-overlapping subregions of the initial representation. For example, a max-pool filter of size 2×2 applied to an input of size 100×100, will slide a non-overlapping window of size 2×2 and extracts the maximum element from the input at each overlap resulting in an output of size 50×50.

The first of the CONV-BN-RELU-MAXPOOL layers applies 32 2D-*CONV* filters of size 3×2, and down samples the feature sizes by 5 and 2 over temporal and spatial (EMG channels)

domains. Similarly, the filter sizes and number of filters of the other layers is depicted in Fig.4.6. The second and third layers down-sample by (4×2), and (2×2) over time and space. Thus, the final output of the encoded representation is of dimensions $25 \times 1 \times 256$. The decoder processes this encoded data to obtain finger joint angles.

Residual Blocks: A natural question to ask is: *Why not increase the depth of the network to extract stronger feature representations?* Unfortunately, deeper networks are harder to optimize and they also pose challenges in convergence. ResNets [190] proposed a revolutionary idea of introducing skip connections between layers so as to balance this tradeoffs between stronger feature representations and convergence. The skip connections, also called as residual connections provide shortcut connections between layers as shown in the middle of the network in Fig. 4.6. Suppose, y, and x, denote the intermediate representations at different layers in the network, with y being deeper than x with a few layers in between. Then, the skip connections are denoted by the below equation.

$$y = f(x, W_l) + x \tag{4.8}$$

$f(x, W_l)$ denotes the intermediate layers between x, and y. Because of the existence of a shortcut path between y and x, the representation at x is directly added to $f(x, W_l)$. Therefore, the network can choose to ignore $f(x, W_l)$, and exploit the shortcut connection $y = x$ to first learn a basic model. As the network continues to evolve, it will exploit the deeper layers $(f(x, W_l))$ in between shortcut connections to learn stronger features than the basic model. As shown in Fig. 4.6, we incorporate ResNets in between the encoder and decoder part of the network. As evaluated in Sec. 4.5, this design choice plays a critical role in achieving a high accuracy.

Decoder: The decoder maps the encoded representations into 3D hand poses. The decoder uses upconvolutional layers to upsample and increase the size of the encoded representation to match the shape of the output. The decoder network consists of a series of CONV-BN-RELU-UPSAMPLE layers. Each such layer consists of following sub-layers. (i) The CONV layer tries to begin making progress towards mapping the encoder representations into joint angles. The job of (ii) BN sub-layer, and (iii) RELU activation sub-layer is similar to their roles in the encoder. (iv) The upsampling sub-layer's job is to increase the sampling rate of the feature representations. Upsampling (with nearest neighbor interpolation method [201]) across multiple layers will gradually increase the size of the compact encoder features to match the size of the output.

The size and number of conv filters in the decoder at each layer is shown in Fig. 4.6. The

three layers of the decoder upsample by factors of (5×4),(4×2),(2×2) respectively on temporal and spatial domains thus matching the output shape of 1000×16 at the last layer. Finally, the decoder output is subject to a Mean Square Error (MSE) loss function as elaborated next to facilitate training.

Loss Functions and Optimization: In all equations below, $\hat{\theta}$ denotes the prediction by the ML model, whereas θ denotes the training labels from a depth camera (leap sensor [30]).

$$loss_{mcp,f/e} = \sum_{i=1}^{i=4} (\hat{\theta}_{i,mcp,f/e} - \theta_{i,mcp,f/e})^2 \tag{4.9}$$

$$loss_{pip} = \sum_{i=1}^{i=4} (\hat{\theta}_{i,pip} - \theta_{i,pip})^2 \tag{4.10}$$

$$loss_{mcp,a/a} = \sum_{i=1}^{i=4} (\hat{\theta}_{i,mcp,aa} - \theta_{i,mcp,aa})^2 \tag{4.11}$$

The above equations capture the MSE loss in prediction of joint angles of MCP (flex/extensions and adduction/abduction), and PIP joints of the four fingers.

$$\begin{aligned} loss_{thumb} = (\hat{\theta}_{th,mcp,aa} - \theta_{th,mcp,aa})^2 + \\ (\hat{\theta}_{th,mcp,f/e} - \theta_{th,mcp,f/e})^2 + (\hat{\theta}_{th,tm,aa} - \theta_{th,tm,aa})^2 + \\ (\hat{\theta}_{th,tm,f/e} - \theta_{th,mcp,f/e})^2 \end{aligned} \tag{4.12}$$

The above equations capture the MSE loss in the MCP and TM joints of the thumb.

$$loss_{smoothness} = ||(\nabla\hat{\theta}_t - \nabla\hat{\theta_{t-1}})||_2^2 \tag{4.13}$$

The above equation enforces constant velocity smoothness constraint in the predicted joint angles where θ_t above is a representative vector of all joint angles across all fingers at a time step t.

The overall loss function is given by the below equation.

$$\begin{aligned} loss = loss_{mcp,f/e} + loss_{mcp,aa} + \\ loss_{pip} + loss_{thumb} + loss_{smoothness} \end{aligned} \tag{4.14}$$

Note that the loss function does not include θ_{dip} or θ_{ip} because we compute them directly from anatomical constraints: Equations 4.1, 4.2.

Finger motion range constraints: As described above, each finger joint has a certain range of motion for both flex/extensions and abduction/adductions. In order to apply these constraints, we first normalize the predicted output of a joint angle by dividing it by the range constraint (for example, by 90° for θ_{dip}). We then apply the bounded ReLU activation (bReLU) function [202] to the last activation layer in our network. The bReLU adds an upper bound to constrain its final output. The bReLU outputs are multiplied again with their range constraints such that the unit of the output is in degrees. The bReLU, in conjunction with other loss functions based on temporal constraints facilitates predicting anatomically feasible as well as temporally smooth tracking results.

4.4.2 Transfer Learning with Semi Supervised Domain Adaptation

For the encoder-decoder model proposed above, training separate models for each user will be burdensome. Therefore, we explore domain adaptation strategies to *pretrain* a model with one (*source*) user and *fine-tune* it to adapt to new users with low training overhead.

Transfer-learning based domain adaptation is popular in vision and speech processing. For example, AlexNet model [203] pretrained on ImageNet database [204] was fine-tuned for classifying images in medical domain [205], remote-sensing [206] and breast-cancer [207]. Similarly, a pre-trained BERT language model [208] was fine-tuned for tasks such as text-summarizing [209], question answering [210] etc. This significantly reduces the burden of training for a new task. In a similar spirit, we use pretrained model from one user and fine-tune it for a different user to significantly decrease the training overhead (Fig. 4.14(a)) without losing much of accuracy.

At a high level, we exploit domain adaptation at the Batch Normalization (BN) layers. Given the sufficient success of BN layers in accelerating convergence by minimizing *covariate shift* [198] with a relatively fewer number of parameters, we exploit them towards domain adaptation as well. The success of this approach has already been shown in other domains such as computer vision [211, 212].

Our domain adaptation process is performed as enumerated below: (i) We generate a model for one user by extensively training the model with labelled data from that user – known as the *pretrain*ed model. (ii) We collect small training data with only few labels from the new (*target*) user. Instead of developing the model for the *target* user from scratch, we initialize the model weights to be same as the *pretrain*ed model. (iii) We make all layers in the model untrainable except the Batch Normalization (*BN*) layers. Using the few labels from the *target* user, we update the BN layers to minimize the loss function. This is called *fine tuning*. The model thus generated will be used for making inferences on the *target* user.

Finetuning the BN layers help with domain adaptation because of their ability to contain wide oscillations in the distributions of input fed from one layer to the next. Given the sufficient success in BN layers (with only a few parameters) for accelerating convergence by minimizing *covariate shift* [198], we exploit them towards domain adaptation as well. The BN layers will learn to sufficiently transform the distribution from *target* user to a distribution of the *source* user on which the model is *pretrained* on. If successful, the *pre-trained* model from the *source* user can be used for performing inferences on the target user with the *finetuning* steps discussed here. As discussed in Section 4.5, this results in reduction of training overhead on the *target* user by an order of magnitude.

4.4.3 RNN Architecture

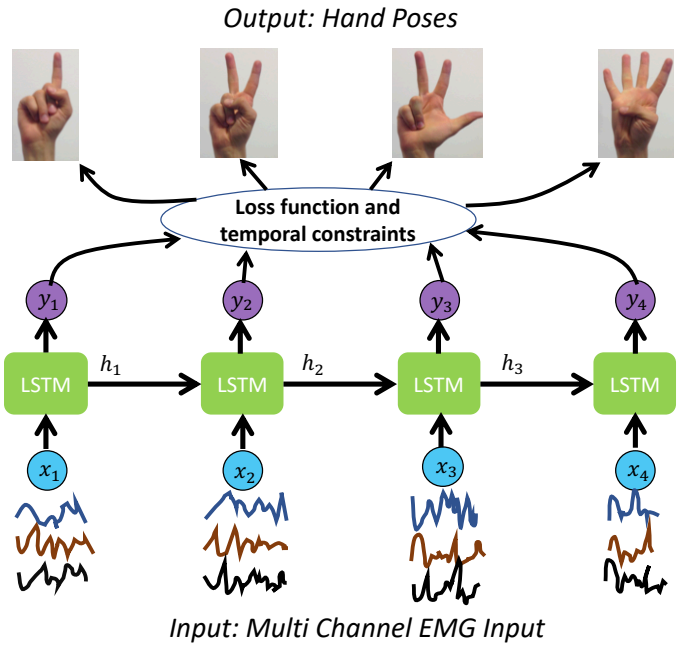


Figure 4.7: RNN alternative explored in this paper.

The encoder-decoder model proposed above has a holistic view of a relatively long interval (5s) of sensor data, and thus can exploit complex spatio-temporal relationships. However, in order to ensure real-time performance with this model, we need to constantly process previous 5s of data at any given instant. Although this can ensure real-time performance, the power consumption can be higher due to redundant computations. Therefore, we explore an alternative model with Recurrent Neural Networks (RNN) to obtain real-time performance without redundant computation.

Our model is presented in Fig.4.7. The generated EMG sensor data is not only dependent

on muscle contractions to maintain the current finger pose but also dependent on the force exerted in the muscles to move the fingers to a new position. Such temporal dependencies can be systematically modeled with a recurrent neural network (RNN). Each RNN unit accepts as inputs one sample of an eight channel EMG data as well as previous hidden state. In particular, we use the Long Short Term Memory (LSTM) variant of RNN because of its ability to handle vanishing/expanding gradients [213] and selectively forgetting/remembering features from past. It outputs an $\mathbb{R}^{16}$ dimension finger joint angles and a new hidden state to be used as input in the next iteration of the RNN unit.

During training, the outputs are subjected to MSE loss functions, as well as temporal constraints identical to ones used in encoder-decoder architecture. We use truncated backpropagation through time (TBPPT [214]) in training with a truncation of 64 time units.

4.5 Performance Evaluation

Our experiments are designed to comprehensively test the robustness to sensor positions, usability, and accuracy of *NeuroPose* over users, joint angles etc. We also compare various ML models, overall training cost as well as perform system level measurements for efficiency of implementation on smartphones.

4.5.1 User Study

We conduct a study with 12 users (8 males, 4 females). The users are aged between 20-30, and weigh between 47-96kgs.

Data Collection Methodology: The users wear the Myo armband as shown in Fig.4.5 on the left hand in a position where it fits naturally, with channel number 4 on top. The users were then instructed to perform random finger motions that include flexing or extending of fingers as well as abduction or adduction thus incorporating all range of possible hand poses. Under the guidance of a study team member, we let the users practice finger motions before the study to ensure that the user moves all fingers over the entire range of motion. This ensures good convergence of the ML models as well as generalizability to arbitrary finger motions. There are no discrete classes of gestures. The motion patterns are entirely arbitrary thus making the data collection easier.

Labels for Training and Testing: The collected data includes 8 EMG channels from the Myo sensor as well as the fingers' 3D co-ordinates and joint angles captured by leap motion sensor [30]. While the Myo sensor provides EMG data for 3D pose tracking, the leap sensor data serves as the ground truth for validation as well as provides labels for training *NeuroPose*'s

ML models. These labels include joint angles for each finger. The EMG and leap data were synchronized using Coordinated Universal Time (UTC) timestamps. Since *NeuroPose* performs continuous finger tracking instead of identifying discrete gestures, we use MSE (instead of cross-entropy) between predicted joint angles (from Myo) and leap (ground truth) for training and testing.

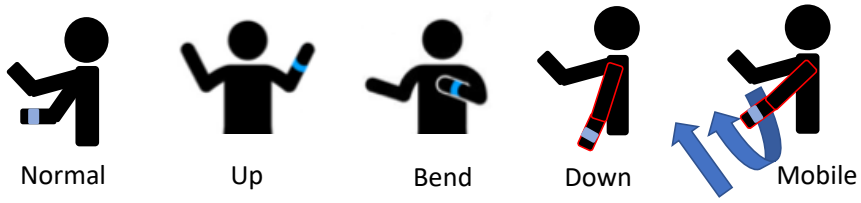


Figure 4.8: Wrist Configuration Map

Training Data Collection : Each user participates in 12 separate sessions with each session lasting for 3 minutes, with sufficient rest between sessions. For the first 5 sessions, both the sensor position and the wrist position are not changed (wrist maintained at the "normal" position depicted in Fig. 4.8). For the each of the last 6 sessions, we remove and remount the sensor. For the 6^{th} session, we let the user place the wrist still in the normal position. However, for the last 6 sessions, we let the user place the wrist in 4 different configurations (*up*, *down*, *bend*, *mobile*) as indicated in Fig. 4.8. In the *mobile* configuration, the wrist was moved up and down including rotations of the wrist within the tracking range of the leap sensor. Users perform *up*, *down*, *bend*, *down*, *up* for sessions 7-11 respectively. For the last session, the users perform the *mobile* configuration. The position of the leap sensor was adjusted using a tripod so that it can capture the ground truth. This data is used for developing two kinds of models. (i) **User-dependent model:** A model for each user that requires 900 seconds of training data from the first 5 sessions of that user. (ii) **Model with domain adaptation:** A model for each user where a pre-trained model from a different user is taken and fine-tuned using techniques in Section 4.4.2 such that only a small fraction (90 seconds) of user-specific training data is used for developing a model for the user. (iii) **Model without domain adaptation or user-independent model:** Here, we use the trained model from one user directly to perform inferences on a new user without any training data from the new user. (iv) **Multi-user model:** This is also a user-independent model. Here, we train a model based on training data from multiple users. The trained model is directly used for inferences on a new user without any training data from the new user.

Test Data: Using the models developed above, we evaluate the joint angle prediction accuracy over test cases that include the last 6 sessions where (i) The sensor has been removed and remounted on the user's arm (ii) The wrist position is completely different from the one

used to train the models.

4.5.2 Implementation

NeuroPose is implemented on a combination of desktop and smartphone devices. The ML model is implemented with TensorFlow [215] packages and the training is performed on a desktop with Intel i7-8700K CPU, 16GB RAM memory, and Nvidia GTX 1080 GPU. We use the Adam optimizer [216] with a learning rate of 1e-3, β_1 of 0.9 and β_2 of 0.999. To avoid over-fitting issues that may happen in the training process, we apply the L2 regularization [217] on each CONV layer with a parameter of 0.01 and also add dropouts [200] with a parameter of 0.05 following each RELU activations. Once a model is generated from training, the inference is done entirely on a smartphone device using TensorFlowLite [10] on a sony xperia z3 with a Quad-core 2.5 GHz Krait 400 CPU.

4.5.3 Performance Results

If not stated otherwise, the reported results are under the following conditions: (i) Averaged across the test cases where the sensor has been removed and remounted, as well as the wrist position is different from one used during training. (ii) Uses the *model with domain adaptation* as described above that requires approximately 90 seconds of training data from each user. The user-independent case is separately evaluated under *model without domain adaptation* (Fig. 4.11(c)), and *multi-user models* (Fig. 4.10(a)). The performance of user-dependent models are also shown separately (Fig. 4.14). (iii) Combines the Encoder-Decoder architecture (including ResNets) in Section 4.4.1 with semi supervised domain adaptation in Section 4.4.2 because that is the best performing design with minimal training overhead for different users. The RNN design presented in Section 4.4.3 is evaluated separately. (iv) The errors reported are for flex/extension angles as they are prone for more errors with a high range of motion. The errors for abduction/adduction are discussed separately (Fig. 4.12(b)). (v) The error bars denote the 10^{th} percentile and the 90^{th} percentile errors.

Qualitative Results: A short demo is provided in this url [3]. Fig. 4.9 shows qualitative results from *NeuroPose*. The predicted hand pose matches closely with reality for a number of example applications including holding virtual objects, ASL signs, pointing gestures etc. Figs. 4.9(a) to (c) include static positions, whereas Figs. 4.9(d) to (g) capture the pose while in motion. Fig. 4.9(h) is an example of an error case. Our inspection of error cases suggests that in most cases, *NeuroPose* is following the trend in the actual hand pose, albeit with a small delay. This delay introduces errors. Another observation is that the ground truth's (leap

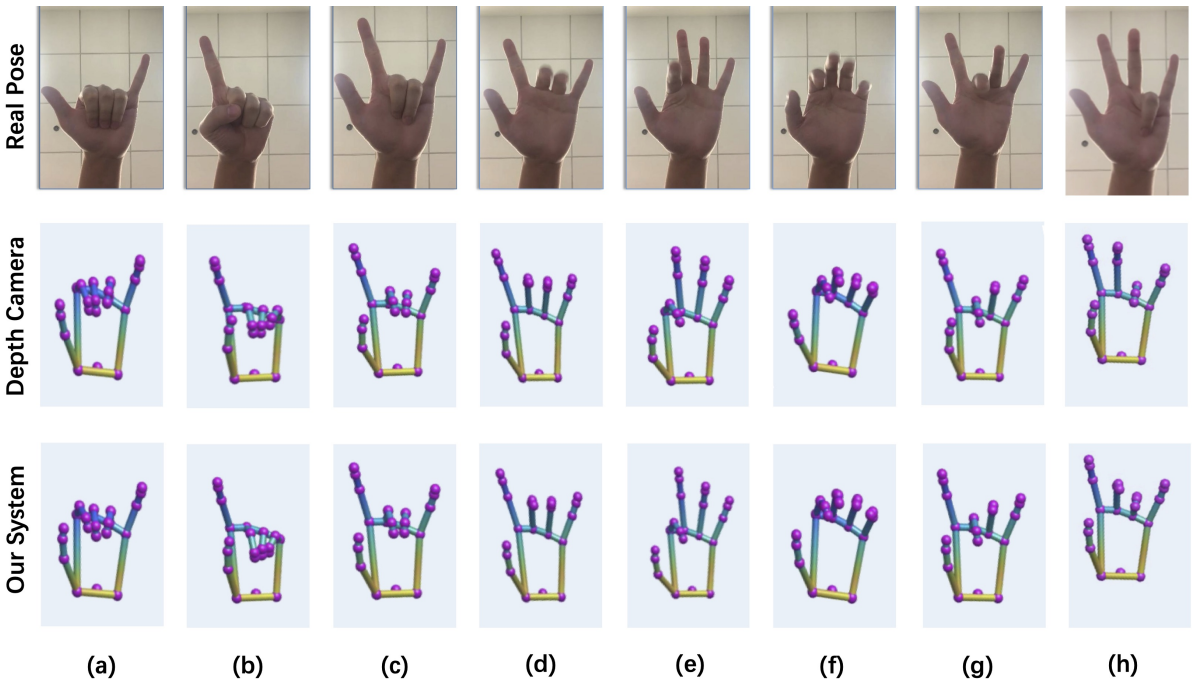


Figure 4.9: Comparison of pose tracking results between depth camera (ground truth) and *NeuroPose*.

depth sensor) detected range of motion for thumb is slightly limited. Extreme thumb motion between Figs 4.9(a) and (b) causes only a small deviation of the thumb in the leap sensor results. Nevertheless, *NeuroPose*'s prediction of thumb angles match closely with the leap sensor (ground truth).

Accuracy over Users: Fig.4.10(a) shows the breakup of accuracy across users over all joint angles. Although the direct use of a model trained from 11 users (multi-user model) and tested on a new user (without domain adaptation) performs reasonably well with a median error of 9.38° degrees, the $90\% - ile$ errors can be huge.

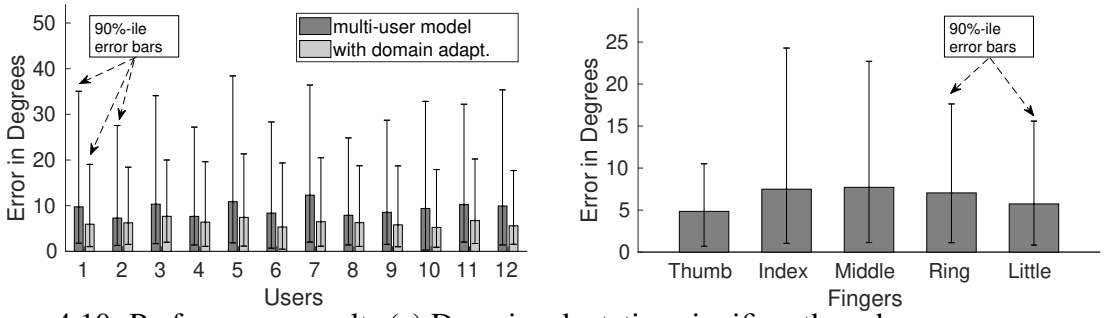


Figure 4.10: Performance results (a) Domain adaptation significantly reduces errors over users (b) Accuracy is consistent across fingers.

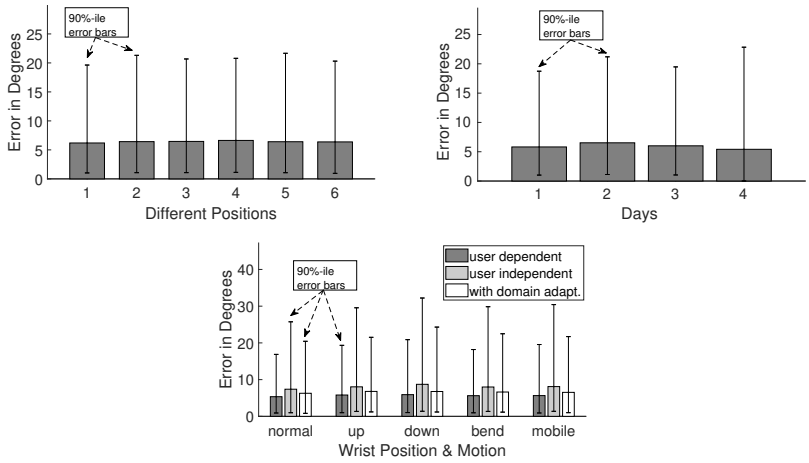


Figure 4.11: Robustness to positions (a) change in sensor position within a day (b) across days (c) change in wrist positions

On the other hand, semi-supervised domain adaptation techniques not only decreases the median error to 6.24° but also cuts down 90%-ile tail error bars dramatically. The accuracy is robust with diversity in users, their body mass indices, gender etc.

Robustness to Natural Variations in Sensor Position and Orientation: We evaluate robustness to natural variations in sensor position by removing the sensor and remounting. Fig. 4.11(a) shows the accuracy when the sensor position was changed 6 times by removing and remounting (these are the last six sessions of data collection phase). Evidently, the accuracy is consistent across all positions. In addition, we followed up with all the users over 4 more days to evaluate the robustness over time, temperature, humidity etc. Fig. 4.11(b) shows the accuracy when the sensor position change happens across multiple days (with a random wrist position). The model that was initially trained continues to provide consistent accuracy over time thus enhancing the usability of *NeuroPose*. We hypothesize that the robustness comes due to three reasons (i) With a snugly fit sensor, its position and orientation changes only by a few mm. The "channel number four" among the 8 EMG channels is clearly marked on the sensor making it easier for the user to maintain the same orientation across multiple sessions of wearing. (ii) Based on the muscle structure map in Fig. 4.3 which extends from elbow to wrist, the relative positions of the target muscles and the sensor changes only slightly. (iii) The Myo sensor warms up the muscles to ensure good contact with electrodes [218], we believe this helps in robustness for temperature change over days

Robustness to Wrist Position and Mobility: Fig. 4.11(c) shows how the accuracy is consistent despite changes in wrist positions. *NeuroPose* can track finger motions accurately even when the wrist is moving. We hypothesize that regardless of the state of the wrist, ML

algorithms always track the muscles responsible for finger motion. The muscles activated for finger motions is independent of the state of the wrist.

Accuracy over Fingers: Fig.4.10(b) provides a breakup of joint angle accuracy over various fingers. For each finger, the accuracy is computed over $\theta_{mcp,f/e}, \theta_{pip}, \theta_{dip}$ angles. For the thumb, the accuracy is computed over $\theta_{mcp,f/e}, \theta_{tm,f/e}, \theta_{ip}$. Overall, the results suggest that *NeuroPose* can track all of the fingers with reasonable accuracy. Although the median error of the index finger is similar to other fingers, one reason why the 90%-ile error is higher could be because the *Extensor Indicis* muscle responsible for index finger motion is a non-surface muscle. Nevertheless, we believe the tracking results of the index finger is promising.

Accuracy over Flex/Extension Joint Angles: Fig.4.12(a) shows the accuracy breakup

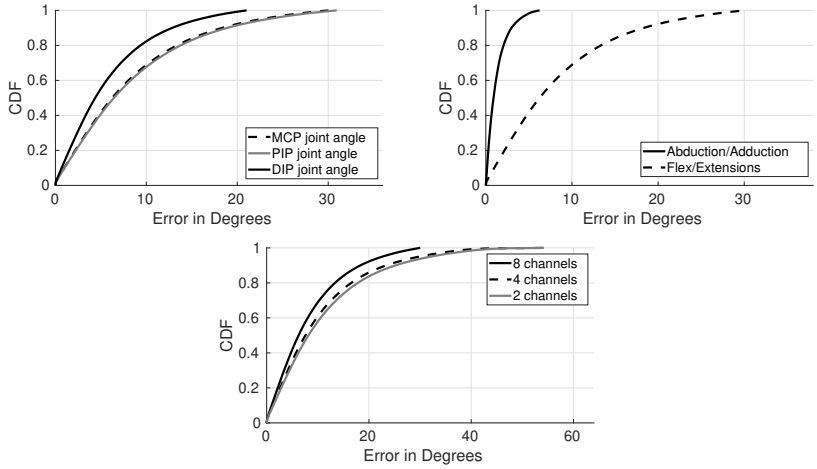


Figure 4.12: (a) Accuracy over MCP, PIP, and DIP joints (b) Accuracy over abduction/adductions and flex/extensions (c) Accuracy vs intrusiveness (number of EMG channels)

between the three flex angles – $\theta_{mcp,f/e}$, θ_{pip}, and θ_{dip}. Evidently, *NeuroPose* maintains similar accuracy for all joint angles. Fig.4.12(b) depicts that the error in abduction/adduction is smaller than flex/extension angles. This is because the range of motion is very limited in abduction/adduction angles.

Intrusiveness and Accuracy Trade-offs: Fig.4.12(c) illustrates the accuracy over number of EMG channels. As expected, the best results are achieved with all 8 channels. However, the error when only using 4 or even 2 channels (shown in Fig. 4.13) offers a reasonable trade-off between accuracy/intrusiveness. Evidently, the median accuracy with 4 and 2 channels is comparable to the case with 8 channels, even though the tail errors are higher. This suggests the promise in further decreasing the intrusiveness of the system.

Training Overhead: Fig.4.14(a) shows the accuracy as a function of amount of training data. Evidently, with domain adpatation strategies proposed in *NeuroPose*, even a small fraction

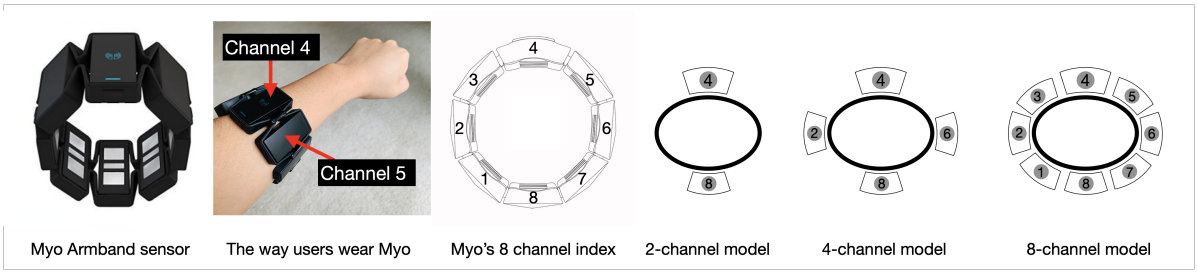


Figure 4.13: 2-channel model and 4-channel model compared to 8-channel model for Myo armband sensor

(1% - 5% or $9 - 45$ seconds) of training data is sufficient to generate a model that is as accurate as a model that uses 90% (or 13.5 minutes) of training data without domain adaptation. This demonstrates the ability in *NeuroPose* to quickly generate a model for a new user with an order of magnitude lesser training overhead than training from scratch.

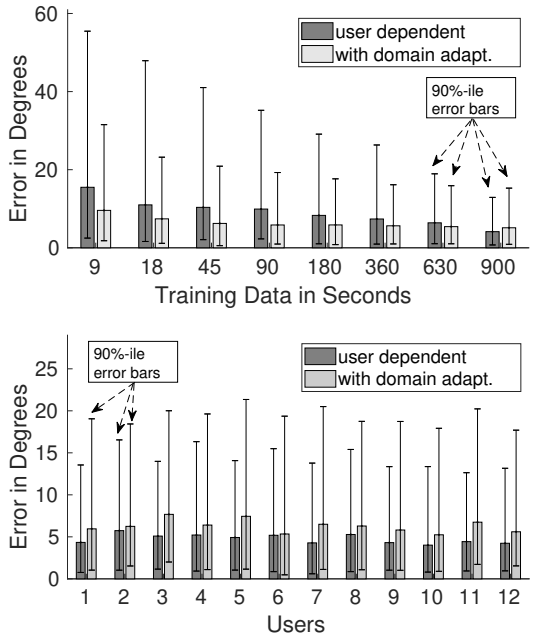


Figure 4.14: (a) Domain adaptation minimizes training overhead by an order of magnitude (b) Performance of domain adaptation is close to user dependent training

User Dependent Training: Although *NeuroPose* performs semi-supervised domain adaptation to generate a model for a new user without extensive training, we evaluate the performance when extensive training is performed for each user to generate her own model. Fig.4.14(b) summarizes the result. User dependent training can improve the median error by $1.52°$, the domain adaptation techniques adopted by *NeuroPose* is close to this performance.

Accuracy Breakup by Techniques, Comparison to Prior Work:

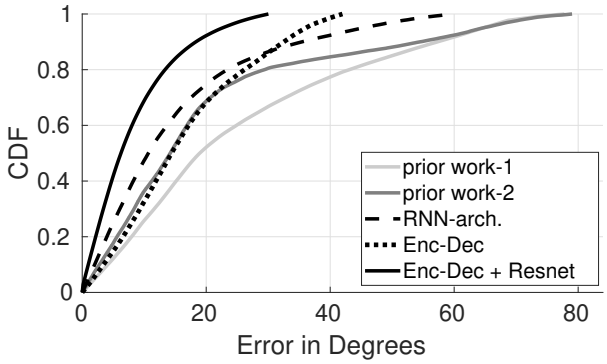


Figure 4.15: Encoder-Decoder-ResNet outperforms other techniques

Fig.4.15 shows the CDF of error comparisons over various techniques and prior work. Prior work-1 [74] includes an LSTM architecture augmented with a Gaussian process for modeling the error distribution and performs hand pose tracking over a specific set of seven discrete gestures. Prior work-2 [75] uses a RNN architecture with a Simple Recurrent Unit (SRU) and extends [74] with experiments over six specific wrist angles.

Although the algorithms are trained and tested over discrete gestures in the original works, our implementation of these algorithms over arbitrary finger motion gives a median error of $18.95, 14.18$ respectively, with a long tail reaching upto $57.31, 54.49$ in the 90%-ile respectively. On the other hand, our LSTM architecture that imposes temporal smoothness constraints across multiple handposes brings down the median accuracy to 10.66, and the 90%-ile accuracy to 35.45. The basic Encoder-Decoder architecture performs slightly better with a median accuracy of 14.40 and a 90%-ile accuracy of 32.52. Finally, *NeuroPose* which exploits deeper features by combining ResNets with Encoder-Decoder architecture outperforms the other techniques dramatically both in the median case and in the tail. The median accuracy is 6.24 and a 90%-ile accuracy is 18.33.

Latency Profiling: The encoder-resnet-decoder model takes 5 second sequence of EMG data as input. The inference latency of processing each 5s of data using TensorflowLite is roughly 0.101 second. At each instant, by processing the previous 5s of data as input, the model can provide an output in 0.101 seconds, thus ensuring real-time performance. This will incur a cost of redundant processing to provide real-time performance – we will discuss the tradeoffs (Fig. 4.16(b)(c)). Furthermore, Fig. 4.16 (a) depicts the average per-sample processing latency of different techniques – LSTM, encoder-decoder and encoder-resent-decoder (NeuroPose) – for relative comparison. The LSTM has a higher processing latency due to the sequential nature of the model with strong dependencies on previous hidden states. In contrast, the encoder-decoder models can exploit parallelism over the entire 5s segment of data.

Power Consumption Analysis: The MYO sensor consumes $40mW$ of power [219], which lasts a day of constant usage. For profiling the energy of the TensorflowLite model, we use Batterystats and Battery Historian [220] tools. We compare the difference in power between two states (i) The device is idle with screen on. (ii) The device is making inferences using TensorflowLite model. The idle display-screen on discharge rate 4.97% per hour while the discharge rates for various models is shown in Fig. 4.16 (b). The power consumption is very low. Since the encoder-resnet-decoder processes data in chunks of 5s, it will incur a delay of atleast 5s if we process the data only once in 5s. Towards making it real-time, we make a modification where at any given instant of time, previous 5s segment of data is input to the network to obtain instantaneous real-time results. This provides real-time tracking at the expense of power.

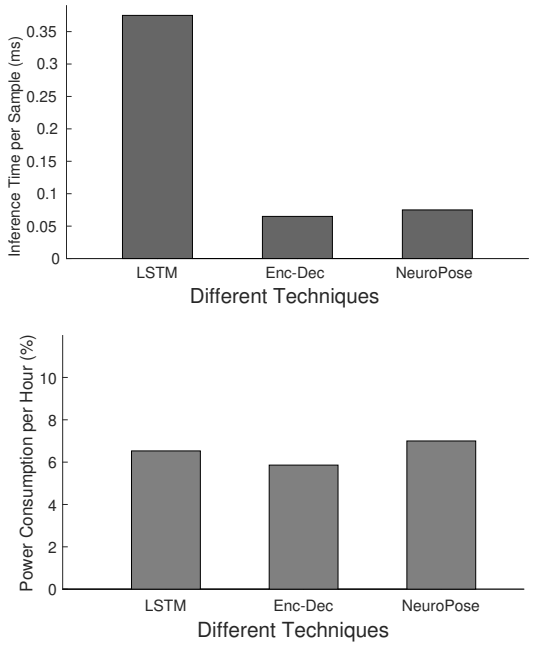


Figure 4.16: (a) Latency comparison (b) Power consumption analysis

4.6 Related Work

Vision: Finger motion can be captured by depth cameras like kinect [29] and leap [30] sensors. However, advances in machine learning, availability of large training datasets have enabled precise tracking of finger motion even from monocular videos that do not contain depth information [31, 32]. While such works are truly transformative, we believe wearable based solutions have benefits over vision based approaches which are susceptible to occlusions, lighting, and resolution. In addition, wearable devices offer ubiquitous solution with continuous

tracking without the need of an externally mounted camera. Most recently, FingerTrak [33] has innovatively designed wearable thermal cameras to track 3D finger motion. However, tracking may not be robust under changes in background temperature as well as motion of wrist (due to shift in camera positions). In contrast, *NeuroPose*'s EMG sensing is robust to background conditions and wrist motion.

Sensor Gloves: Gloves with embedded sensors such as IMU, flex sensors, and capacitative sensors have been used for finger pose tracking in a applications like sign language translation, gaming, HCI etc [15]. Work in [34] tracks hand pose using an array of 44 stretch sensors. Works [35, 36] extracts hand pose using gloves embedded with 17 IMU. Flex sensors have been used in commercially available products such as CyberGlove [37], ManusVRGlove [38], 5DT Glove [39] etc. However, wearing gloves in hands may hinder dexterous/natural hand movements [40].

IMU and wrist bands: IMU and WiFi sensors have been used in a number of localization and human body tracking projects [56, 221, 222]. IMU, WiFi, and Acoustics have also been extensively used for hand gesture recognition. [64, 136, 141, 223, 224]. uWave [142] uses accelerometers for user authentication/interaction with mobile devices. FingerIO [143], FingerPing [112] use acoustics for finger gesture detection. Capband [108] uses capacitative sensing for recognizing 15 hand gestures. In contrast, NeuroPose develops algorithms for generic finger motion tracking. Specifically while prior works can only distinguish multi-finger gestures, *NeuroPose* performs free form 3D finger motion tracking. AuraRing [168], a recent work, tracks the index finger precisely using a magnetic wristband and ring on index finger. In contrast, *NeuroPose* tracks all fingers.

ElectroMyoGraphy: EMG based gesture tracking is an active area with decades of research. Prior works perform classification of discrete hand poses [71–75] or tracking of a predefined sequence of hand poses [74, 75] with a combination of deep learning techniques based on CNN, RNN etc. Works [80] can track joint angles for arbitrary finger motion, but requires a large array of over 50 EMG sensors placed over the entire arm. Work in [81] tracks joint angles using EMG sensors but only for one finger. In contrast to these works, *NeuroPose* tracks continuous finger joint angles for arbitrary finger motions with only sparse EMG sensors.

4.7 Discussion and Future Work

Unsupervised Domain Adaptation: *NeuroPose* only needs $90s$ of training samples from a new user to customize a pretrained model to the user. However, we will explore unsupervised domain adaptation to customize a pretrained model without requiring any labelled training

data. Adversarial domain adaptation [225] is of interest. Here, an unsupervised game theoretic strategy is used to transform the distribution of the feature representations from the new user into the distribution of the source user on whom the model was trained. If successful, the model trained on the source user is directly useful for performing inferences on a new user. Similarly, other architectures for learning feature transformations to adapt the feature representations from a source user to a new users have been proposed [226] which are relevant for future investigation.

Prosthetic Devices for Amputees: While the subjects recruited for this study were able-bodied individuals, we will consider design and evaluation of *NeuroPose* for amputees for future work. In particular, given prior research on *mirrored bilateral training* approach [22, 81, 82], we believe there is promise.

Tracking Fingers while Holding Objects in Hand: When holding an object, signals from certain muscles that support strength will interfere with muscles responsible for finger motion. While we believe there are enough applications in augmented reality and prosthetics where a user does hold an object, we will carefully refine *NeuroPose*'s algorithms to minimize the interference from additional muscle signals when a user is holding an object.

4.8 Conclusion

This paper shows the feasibility of 3D hand pose tracking using wearable EMG sensors. A number of applications in Augmented Reality, Sports Analytics, Healthcare, and Prosthetics can benefit from fine grained tracking of finger joints. While the sensor data is noisy and involves superimposition of signals from different fingers in complex patterns, we exploit anatomical constraints as well as temporal smoothness in motion patterns to decompose the sensor data into motion pattern of constituent fingers. These constraints are incorporated in an encoder-decoder machine learning model to achieve a high accuracy over diverse joint angles, different type of gestures etc. Semi supervised adaptation strategies show promise in adapting a pretrained model from one user to a new user with minimal training overhead. Finally, the inference runs in realtime on a smartphone platform with a low energy footprint.

Chapter 5
Mirrored Bilateral Training

5.1 Introduction

NeuroPose shows the feasibility of 3D hand pose tracking using wearable EMG sensors, in addition to that, a unique motivation for ElectroMyoGraphy (EMG) based tracking over vision and other wearable systems (such as IMU) is that EMG signals from amputees can be used for controlling prosthetic limbs with potential to significantly improve their accessibility needs, the feasibility of which is shown in prior work for index finger motions, wrist motions, gestures etc [22, 81]. However, generating training data can be a challenge for amputees with missing fingers. Our preliminary result in Section 5.4 discusses a *mirrored bilateral training* approach for generating training data for amputees for 3D finger motion tracking. The results are promising with applications in development of prosthetic devices with finer control. The overall system accuracy is robust to natural variation in sensor mounting positions as well as changes in wrist positions of users. Performance comparison across both low-end and more recent smartphone platforms demonstrates a competitive performance across a wide spectrum of devices. Furthermore, we show the applicability of *NeuroPose* in real world use cases such as finger-spelling classification in American Sign Language (ASL) .My main contributions in this chapter includes evaluation of a *mirrored bilateral training* [22] scheme with a potential future application for developing prosthetics for amputees with missing fingers.

5.2 Background

An important application of EMG devices is in developing prosthetic devices for amputees with missing fingers. However, because of missing fingers, it is non-trivial to generate training data that maps EMG signal pattern into corresponding 3D joint angles of various fingers.

Towards handling this challenge, we explore a *mirrored bilateral training* [22] scheme. In this subsection, we introduce the biological foundations of *mirrored bilateral training* as well as provide high level details on exploiting this opportunity for generating training data for amputees.

A unilateral motion such as a motion with the right hand induces involuntary muscle activation in the contralateral part of the human body such as the left hand. This is called as *mirrored bilateral motion* and the corresponding activity in electromygraphy signals is called as *mirrored electromyography* (MEMG). MEMG has been consistently observed in both healthy and pathological cases over a number of simple and complex motor activities in daily life. This is known to happen because of a motor overflow that causes the involuntary muscular activity due to interhemispheric communication within the brain during motion activities [227]. Such interhemispheric communication leads to bilateral activation of motor relevant brain regions. Although the evolution in humans have gone through an ontogenetic learning process that decouples both hands to be independent of each other, the MEMG activity is said to be a remnant of the basic mirror movement mode of the central nervous system [228]. This facilitates mirrored motions under voluntary setting where both hands can move synchronously in nearly identical fashion. As elaborated later in this section, *NeuroPose* will exploit this property in an application for developing prosthetic devices for amputees.

A transradial amputation is one where a part of the arm is missing below the elbow beyond a certain point along the radial bone. Such an amputation might occur because of a number of reasons including injury, tumor, frostbite, infection etc. A number of prosthetic devices have been proposed for such cases. This includes cosmetic prosthetic devices which do not move but used solely for the purpose of appearance [229]. On the other hand, a body powered prosthesis is attached to the body by a series of wires [230]. Moving the body in different ways will move the prosthetic device for performing different activities. Finally, a myoelectric prosthesis is the most advanced form of prosthetic device. EMG signals from the brain can be used to control the prosthetic hand resulting in an effect that is similar to a real hand [231].

At a high level, transradial amputees will still retain the neuromuscular structure that is responsible for precise finger motion. Even though the fingers might be missing, studies have shown that the subjects with amputations are capable of generating neuromuscular potentials that is responsible for a particular pattern of finger motion [22, 81, 82, 232]. By identifying the appropriate patterns, an external prosthetic device can be attached to produce those actions thus providing an experience that is close to a real hand.

While mapping the EMG signals to finger patterns for able bodied individuals is easy because machine learning models can be trained to map the EMG signals to finger motions,

the same is not feasible with amputees. The lack of a finger precludes training data that maps the EMG signals to the motion of that finger. One possibility to handle this challenge is to let the amputee emulate a few predefined finger motion patterns (flicking a finger etc), and record the EMG signals to be used for training [82]. However, the action of the amputee might differ from the predefined finger pattern in temporal alignment, bio-mechanical coupling as well as the intensity of force applied, thus resulting in poor quality of training data. *Mirrored bilateral motion* as described earlier can be exploited as an opportunity to handle this challenge. The neural activation patterns are known to be similar in both hands for performing similar finger motion activities [228]. Therefore, a machine learning model trained with the non-amputee hand (without missing fingers) while inducing bilateral activation can potentially be used for performing inferences on the hand with missing fingers (amputated hand). Thus, appropriate control signals can be generated for controlling the prosthetic device attached to the amputated hand. *NeuroPose* exploits this opportunity and provides insights into the feasibility of such a *mirrored bilateral training* approaches for prosthetics capable of performing fine grained 3D finger motion instead of discrete gestures.

5.3 Core Technical Modules

As discussed in Section 5.2, the neuromuscular interactions are such that an amputated hand still preserves the muscular activation and exhibits strong similarities to the muscular activity in the non amputated hand. Therefore, the training data, labels, and models developed from the non amputated hand can be used for performing inferences on the amputated hand. To handle any residual differences in neuromuscular activity between amputated and non amputated hand, we develop an architecture based on representation learning to further improve the accuracy. Fig. 5.1 depicts the high-level architecture of the representation learning framework used in *NeuroPose*. The raw sensor input x^i is first transformed into two variants (x_1^i and x_2^i) based on data augmentation techniques. The transformations add perturbations to the data while still retaining the overall pattern. x_1^i and x_2^i are then fed as input to the neural network that extracts representations h_1^i and h_2^i as shown in the *self-supervised stage* of the figure. Then, the representations are projected into a latent space before comparing them using a *contrastive loss function* that attempts to maximize the similarity between y_1^i and y_2^i while minimizing the similarity between y^k and y^j, where $k \neq j$. Since such a network tries to enforce similarity among representations even though the inputs have been perturbed by data augmentation techniques, it is known to learn efficient representations. Finally, the representations thus learned are fine tuned with labeled data for predicting the 3D finger motion joint angles.

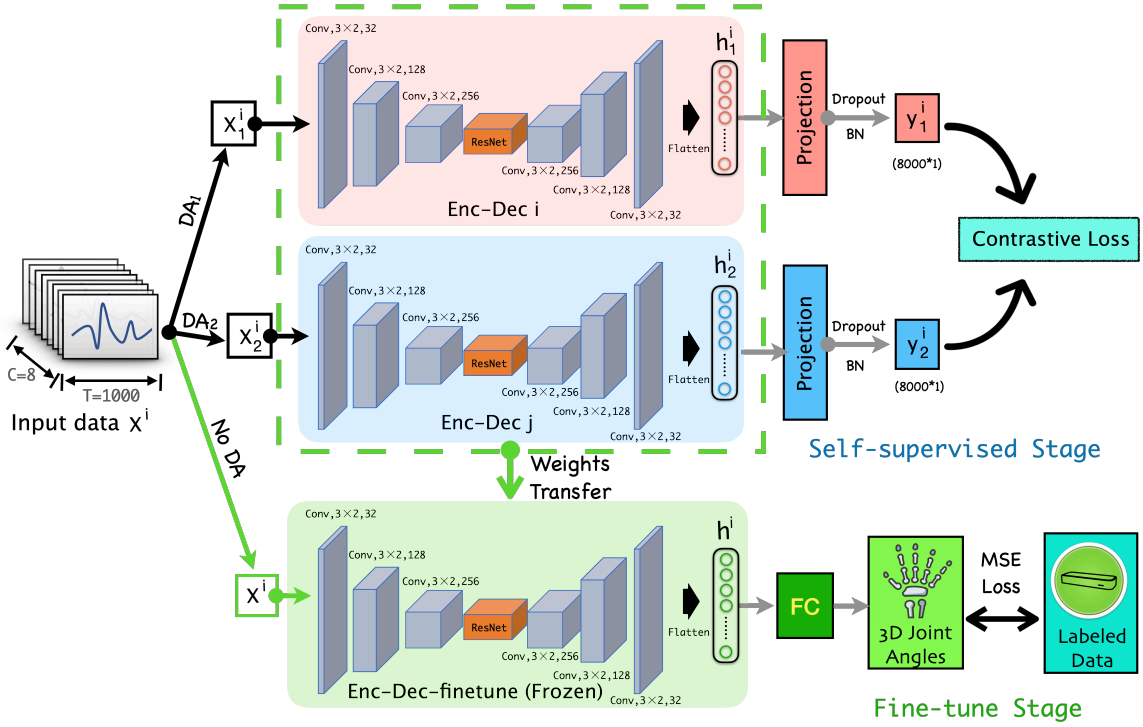


Figure 5.1: Architecture for self-supervised and fine-tuning stages (DA = Data Augmentation)

Evaluated in Section 5.4 using the representations enhances robustness of adaptability of a model trained from a non amputated hand for inference on the amputated hand.

Data Augmentation: Towards learning self-supervised representations, we employ data augmentation techniques to our ML model. This helps avoid overfitting as well as teaches the algorithm to look for stable features that measure similarity. The architecture in Fig. 5.1 needs two data augmentation techniques at a given instant of time. We take a combination of two from the following three techniques: (i) Temporal masking: We mask parts of the input data along time axis so as to help our model in capturing the temporal dependencies in the sensor data. Such strategy is popular in natural language processing such as BERT [208] where words are masked in a sentence to force the language model to predict these words, thus facilitating learning of efficient representations of sentences. Inspired by BERT, we add temporal masking along time as an data augmentation technique. (ii) Noise Addition: We add Gaussian noise to the input data to create augmented versions of the sensor data. Such a process of adding randomness and enforcing similarity between differently augmented copies of the input will teach the model to look for stable features and also help it to avoid overfitting issues. We believe this also help facilitate the mirrored bilateral training process where the training and test data come from different hands, with potential noise between them. (iii) DTW based data augmentation: The speed of hand motion is one of the metrics that can vary across time and

users. Various parts of the finger motion might be performed at a faster or slower pace. Towards making the ML models robust to such variations, we augment the training data by stretching and compressing different parts of the data using DTW [134] based algorithm with different factors.

Contrastive Loss Function: The encoded representations h are passed through a projection head as shown in Fig. 5.1, resulting in an output $y = p(h)$ where p represents the action of the projection head. We apply the contrastive loss function on y that maximizes the similarity between two differently augmented copies of the same input. The contrastive loss function is applied on y whereas we use representations h in the later phases for prediction of 3D finger motion. The reasons for such a design choice are as follows. (i) Since the contrastive loss function's main goal is to maximize the similarity between differently augmented versions of the same input, it might lose some information during the process. (ii) On the other hand, the encoded representation h is one level before the projection head, and it offers the best trade-off between capturing high-level robust representations without losing much information.

The mathematical form of the contrastive loss function that enforces similarity between differently augmented samples of the same input x^i is given by:

$$\ell^i = -\log \frac{\exp(sim(y_1^i, y_2^i)/\tau)}{\sum_{k=1}^{2N} 1_{i \neq j} \exp(sim(y^i, y^j)/\tau)}, \qquad sim(u, v) = \frac{u^T v}{||u|| \cdot ||v||} \tag{5.1}$$

Here, $(y_1^i$,$y_2^i)$ represents the output of the projection head from Fig. 5.1 that acts on differently augmented versions x_1^i, x_2^i of the same input x^i. Given a batch of N input examples $\{x^i,\ \forall\ i\ \in\ [1, N]\}$, we have $2N$ similarity examples of the form $\{(x_1^i, x_2^i),\ \forall\ i\ \in\ [1, N]\}$. For each similarity pair of the form (x_1^i, x_2^i), we have $2(N-1)$ dissimilar pairs of the form $\{(x^i, x^j), i \neq j\}$. $1_{i \neq j}$ indicates dissimilar pairs when $i \neq j$, and τ denotes a temperature parameter that controls the penalty strength on dissimilar samples. In our experiments, we set τ to 0.2 to encourage the model to have tolerance for similar samples within a batch. Both similar and dissimilar pairs are considered in evaluating the contrastive loss function in Equation 5.1 thus training the network to maximize the similarity between similar pairs as well as minimize the similarity between dissimilar pairs. The similarity metric $sim(u, v)$ is also indicated in Equation 5.1.

Prediction of 3D Finger Joints from Self-Supervised Representations: The representations h learned above based on the architecture in Fig. 5.1 are used for estimating 3D finger joint angles. This is indicated as the *fine-tune stage* in Fig. 5.1. The input EMG data is first

passed through the encoder-decoder which extracts representations h. For predicting the finger joint angles using h, we follow a widely used evaluation protocol which can be used as a proxy indicator for the efficiency of self-supervised learning. Specifically, a simple linear model with two fully-connected layers takes the representations h and predicts joint angles. The weights of the linear model are trained on top of the encoder-decoder network (encoder-decoder's weights are frozen after *self-supervised stage* in Fig. 5.1) in a supervised fashion.

5.4 Performance Evaluation

Mirrored Bilateral Training: The right and left hands are mirror images of each other. Thus, the model built from one hand might be usable for the other hand (discussed in Section 5.2), provided that the EMG channel numbers are replaced by their corresponding mirror images (For example, from Fig.4.5, the mirror of channel 5 is channel 3. The exact mirrors of each EMG channel is illustrated in Fig. 5.2). To validate this, we perform more experiments where users perform arbitrary finger motions with *mirrored bilateral training*. The training data from the left hand was then used for performing inferences on test data from the right hand. The ideas in self-supervised learning from Section 5.3 have been used for processing the sensor data to further reduce the noise due to difference in distribution of data across two hands. Fig.5.3 shows the performance.

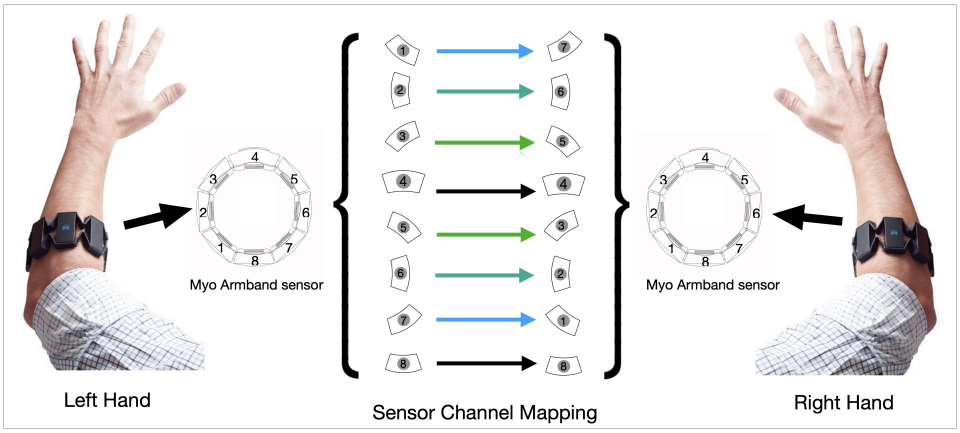


Figure 5.2: Mapping of EMG channels for doing inference on the right hand with training data from left hand

Performance Analysis over an Application in Gesture Recognition: *NeuroPose* performs 3D tracking of finger motion with a number of applications in augmented reality, virtual reality, sports analytics, sign language recognition etc. We evaluate the feasibility of *NeuroPose* over a real world application in recognition of alphabets in American Sign Language (ASL) shown in

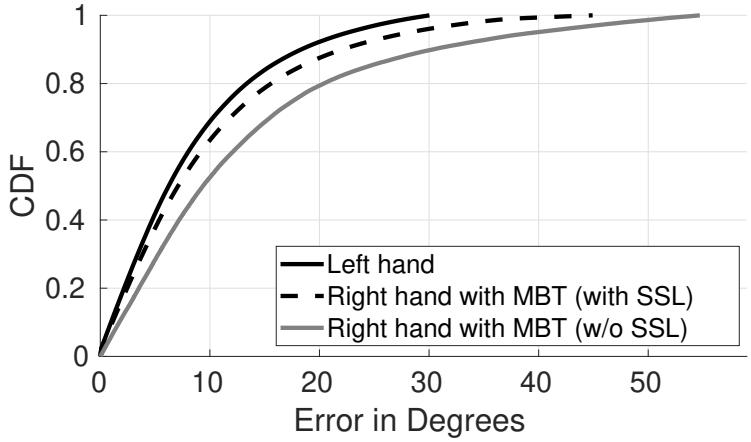


Figure 5.3: Model learnt for the left hand is easily adopted for inferences on the right hand with MBT (MBT means Mirrored Bilateral Training, and SSL means self-supervised learning)

Figure 5.4(a). Four users were recruited to wear the Myo armband and perform the 26 ASL alphabets 10 times each. Training data was collected from one user who performed these same gestures. The classification was performed by comparing the $\mathbf{R^{21}}$ space of joint angles of the test users with that of the training data. The gesture in the training database with the minimum euclidean distance is declared as the inferred gesture. Fig. 5.4(b) depicts the confusion matrix of the classification. Evidently, most gestures are classified correctly with an overall accuracy of 80.22%. Gestures such as A and S are miss classified sometimes because their hand-pose is similar. This demonstrates the feasibility of using *NeuroPose* in real world applications.

Latency Comparison over Phone Models: Fig. 5.5(a) shows the comparison of latency over three different phone models - Sony Xperia Z3, Samsung Galaxy S20, OnePlus 9 Pro. The *NeuroPose* (encoder-resnet-decoder) model takes 5 second sequence of EMG data as input. The inference latency of processing each 5s of data using TensorflowLite on the three different brands of smartphones are 0.067s, 0.012s, and 0.019s respectively. At each instant, by processing the previous 5s of data as input, the model can provide an output in 0.067 seconds even in the worst case of scenario of an older smartphone model (Sony Xperia Z3). This shows how the machine learning models are lightweight thus ensuring realtime performance even on low-end smartphones. However, the encoder-resnet-decoder model will incur a cost of redundant processing to provide real-time performance – we will discuss the tradeoffs (Fig. 5.5(b)(c)). Furthermore, Fig. 5.5 (a) depicts the average per-sample processing latency of different techniques – LSTM, encoder-decoder and encoder-resent-decoder (NeuroPose) across all three brands of smartphone models – for relative comparison. The LSTM has a higher processing latency due to the sequential nature of the model with strong dependencies on

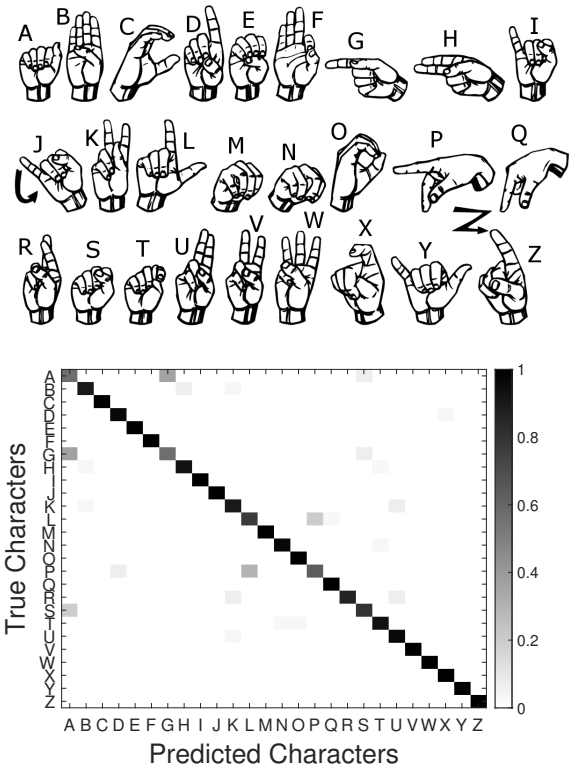


Figure 5.4: (a) ASL alphabets (b) Confusion matrix of *NeuroPose*'s performance in ASL alphabet classification

previous hidden states. In contrast, the encoder-decoder models can exploit parallelism over the entire 5s segment of data. While the increase in latency with LSTM in comparison to encoder-decoder architecture is $\approx 2x$ for newer smartphones (OnePlus 9 Pro, Samsung S20), the increase in latency can be upto 5x for older smartphones (Sony Xperia Z3).

Power Consumption Analysis: The MYO sensor consumes $40mW$ of power [219], which lasts a day of constant usage. For profiling the energy of the TensorflowLite model, we use Batterystats and Battery Historian [220] tools. We compare the difference in power between the following two states across all three smartphone models (i) The device is idle with screen on. (ii) The device is making inferences using TensorflowLite model. The idle display-screen on discharge rate $3-5\%$ per hour while the discharge rates for various models is shown in Fig. 5.5 (b). The power consumption is very low across all brands of phones. Since the encoder-resnet-decoder processes data in chunks of 5s, it will incur a delay of atleast 5s if we process the data only once in 5s. Towards making it real-time, we make a modification where at any given instant of time, previous 5s segment of data is input to the network to obtain instantaneous real-time results. This provides real-time tracking at the expense of power. Depicted in Fig. 5.5 (c) this entails continuous/redundant processing thus increasing the discharge rate to $\approx 20\%$

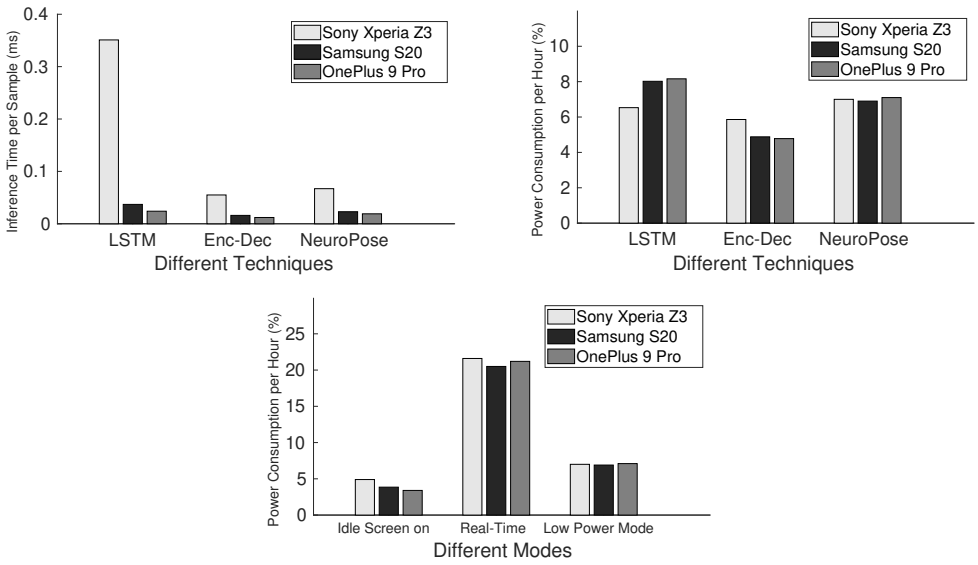


Figure 5.5: (a) Latency comparison (b) Power consumption analysis (c) Power consumption across real-time and energy saving modes

across all phones. The low-power mode trades off real time performance (5s delay) for power savings. Depending on requirements of real-time latency or energy-efficiency, a user can choose between the two modes. The above presented measurements on latency and power consumption show the ability of NeuroPose to perform effectively on a range of embedded smartphone operating systems.

5.5 Related Work

Finger Motion Tracking by Radio Frequency Reflections: Prior works have explored WiFi signals to track motion of hand and classify discrete gestures by using a combination of wireless channel state information (CSI), and doppler shift measurements [45–47]. SignFi [54] is an innovative work that uses wireless channel measurements from WiFi APs for sign language recognition. ExASL [55] tracks point clouds computed from range-doppler spectrum and angle of arrival spectrum of mmWave reflections from the hand. This is used to classify upto 23 discrete hand motion gestures used in ASL. Google Soli [60] exploits reflections from mmWave signals in combination with deep convolutional and recurrent neural networks to track 11 finger motion gestures. In contrast to discrete gesture classification in the above works, *NeuroPose* performs continuous 3D finger motion tracking. In additon, while the above approaches are limited by range of coverage of mmWave and WiFi signals, *NeuroPose* offers a more ubiquitous tracking.

ElectroMyoGraphy: The use of EMG signals for hand pose tracking is an active area with decades of research. Prior works perform classification of discrete hand poses [71–75] or tracking of a predefined sequence of hand poses [74,75] using EMG sensors with a combination of deep learning techniques based on CNN, RNN etc. Work in [76] can classify multi finger gesture sequences using a 4 channel EMG sensor. A number of popular features based on spectral power magnitudes, hudgins' time domain features, correlation coefficients etc have been used in conjunction with SVMs, nearest neighbors, and linear discriminant analysis based algorithms to show the feasibility of gesture classification. Work in [77] uses Myo armband similar to the one used in this paper to classify 5 gestures such as fist, wave-in, wave-out, open, and pinch etc. A shallow feed forward neural network with 3 layers has been used to perform this classification. Work in [78] shows that muscle synergy can be exploited to reduce the dimensions of feature vectors in EMG based gesture classification. Evaluated over five hand activities such as open, close, pinch, valgus, and grasp, the recorded EMG data from the forearm have been compressed using non negative matrix factorization to extract synergistic myo-electrical activities. The compressed feature set has shown to demonstrate a higher recognition rate. Work in [79] uses forearm EMG signals to control a robotic arm. A set of 9 gestures are detected to contral a 6 degree of freedom robotic arm. Ensembled bagged trees, SVM, and neural networks have been used to perform the classification. Works [80] can track joint angles for arbitrary finger motion, but requires a large array of more than 50 EMG sensors placed over the entire arm. Work in [81] tracks joint angles using EMG sensors but only for one finger. In contrast to these works, *NeuroPose* performs accurate tracking of continuous finger joint angles for arbitrary finger motions with only sparse EMG sensors.

Mirrored Bilateral Training: Work in [22] estimates the force on contralateral arm using EMG signals measured from the other arm. A multilayer perceptron (MLP) based algorithm has been used to make the association between EMG signals and the associated force in the arm. Based on several experiments with tens of individuals , this paper shows that an accurate estimation of forces in the contralateral limb can be done based on the EMG signal from the other arm, thus showing promise. Similarly, work in [81] shows the feasibility of estimation of flex and extension joint angles of one finger based on EMG data collected from the other hand. A number of features such as zero crossings, mean absolute value, waveform length, slope changes etc has been applied on EMG data. Furthermore, a state space model with parameters estimated from contralateral arm is used to estimate the joint angle of one finger on the other arm. The results show an estimation error under 1 degree thus indicating sufficient promise. Work in [82] compares training via mirrored EMG from contralateral arm with training by mimicking gestures on the same arm with potential amputation. Evaluated over more than

20 gestures, a better performance is achieved by mirroring on the contralateral arm instead of mimicking with the same arm that may have amputation. The main challenge with mimicking is identified as the inability to estimate force involved in motion as well as misalignment over time with between the imitation and the actual gesture. Work in [83] can perform wrist motion classification using *mirrored bilateral training*. Based on the EMG data from the contralateral arm, and employing techniques based on artificial neural networks for pattern classification, upto 70% in accuracy has been shown in terms of classification of 4-6 wrist motion gestures. All of the above works show promise in the technique of *mirrored bilateral training*. In contrast to these works which either track discrete gestures or continuous motion of one finger, *NeuroPose* shows the feasibility of *mirrored bilateral training* for continuous estimation of 21 degrees of freedom involved in 3D hand pose estimation.

5.6 Conclusion

The feasibility of *mirrored bilateral training* has been shown for 3D finger motion tracking with potential to develop prosthetic devices for amputees in this work. Semi supervised adaptation strategies show promise in adapting a pretrained model from one user to a new user with minimal training overhead. Finally, the inference runs in realtime on a smartphone platform with a low energy footprint.

Chapter 6
ZeroNet

6.1 Introduction

Finger motion tracking enables exciting IoT applications in sports analytics [13], healthcare and rehabilitation [14, 233], sign languages [15], augmented reality (AR), virtual reality (VR), etc. Analysis of finger motion of aspiring players can be compared to experts to provide automated coaching support. In the context of healthcare, finger motion stability patterns are known to be bio-markers for predicting motor neuron diseases [17, 170]. AR/VR gaming as well as precise control of robotic prosthetic devices are some of the other applications that benefit from finger gesture tracking [20, 21].

Motivated by the above applications and coupled by recent innovations in machine learning (ML) and the availability of large scale training data, there is a surge of recent research [31, 32, 150] in computer vision that track accurate finger poses from monocular videos. Given that they do not require depth cameras, the range of applications enabled is far reaching. However, such vision based techniques are affected by issues such as occlusions and the need for good lighting conditions to capture intricate finger motions.

In contrast to vision, the main advantage of wearable IoT devices lies in enabling ubiquitous tracking without external infrastructure while being robust to lighting and occlusions. However, unlike vision, there is a dearth of large scale training data to develop robust ML models for wearable devices. Towards overcoming this gap, this paper presents a system called *ZeroNet*. This system requires zero training overhead for developing robust ML models for finger motion analytics using smart-ring based Inertial Measurement Unit (IMU) sensors. In particular, *ZeroNet* harvests training data from public videos of finger motions and develops ML models that can be used for inferences on smart-rings with IMU sensors.

Such a method of learning from one domain for performing inferences on a different domain has been explored before. Unsupervised domain adaptation [225, 226] can adapt distributions between source (video) and target (IMU) domains such that the model learnt on source domain is used for inference on target domain. However, such techniques are hard to apply to our problem domain since this still requires enough real training data (atleast in unlabelled form) from IMU to achieve sufficient convergence of the domain adaptation process. Furthermore, each user's finger motion pattern as well as natural variations in sensor wearing positions could lead to different distributions in the sensor data [99, 234] thus entailing more training data for each setting. On the other hand, *ZeroNet* performs comparable to models developed with semi-supervised domain adaptation [205, 206] which need partial labelled real IMU data and even outperforms models fully trained on our own real IMU dataset (details in Sec. 6.6). Given lack of large training datasets under diverse conditions for smart-rings, we believe *ZeroNet*'s ability to provide promising accuracy without any training cost is an important first step to bootstrap applications. With enough applications, more data can potentially be generated via crowd-sourcing approaches to further push the accuracy of domain adaptation.

Fig. 6.1 illustrates the architecture of *ZeroNet* with the following sequence of actions. (i) Appropriate sources of publicly available videos (YouTube, ViMeo, Flickr etc.) are first identified as candidates for training data. (ii) Finger locations are then extracted from these videos using computer vision techniques [235,236]. (iii) Appropriate motion metrics that can be captured from IMU (acceleration, orientation etc) are then derived from these finger locations. (iv) The training data thus extracted from videos is further enlarged using data augmentation techniques (introducing variants of rotations, speed of gestures, temporal clipping etc) to create a large and high quality training dataset. (v) Such synthetic datasets are used for training ML models (vi) Finally, the trained models can be deployed directly for inferences on wearable devices with zero training overhead. Inspired by favorable usability reviews of smart-rings in monitoring activity in gym, sleep etc., [129–131] we place a sensor on the finger for gesture inferences (details in Sec. 6.3).

Although in a similar spirit to recent works [102–105] showing the feasibility of harvesting training data from videos for identifying upto ten classes of human activities, *ZeroNet* differs from the above works in following ways: (i) Shows the feasibility of harvesting training data from videos for a gesture recognition problem involving intricate finger motions. (ii) The harvested training data from videos is combined with data augmentation techniques to enable better generalizability of ML models. (iii) Shows the ability of recognition over 50 classes – a five fold higher number of classes than prior work extracting training data from public videos.

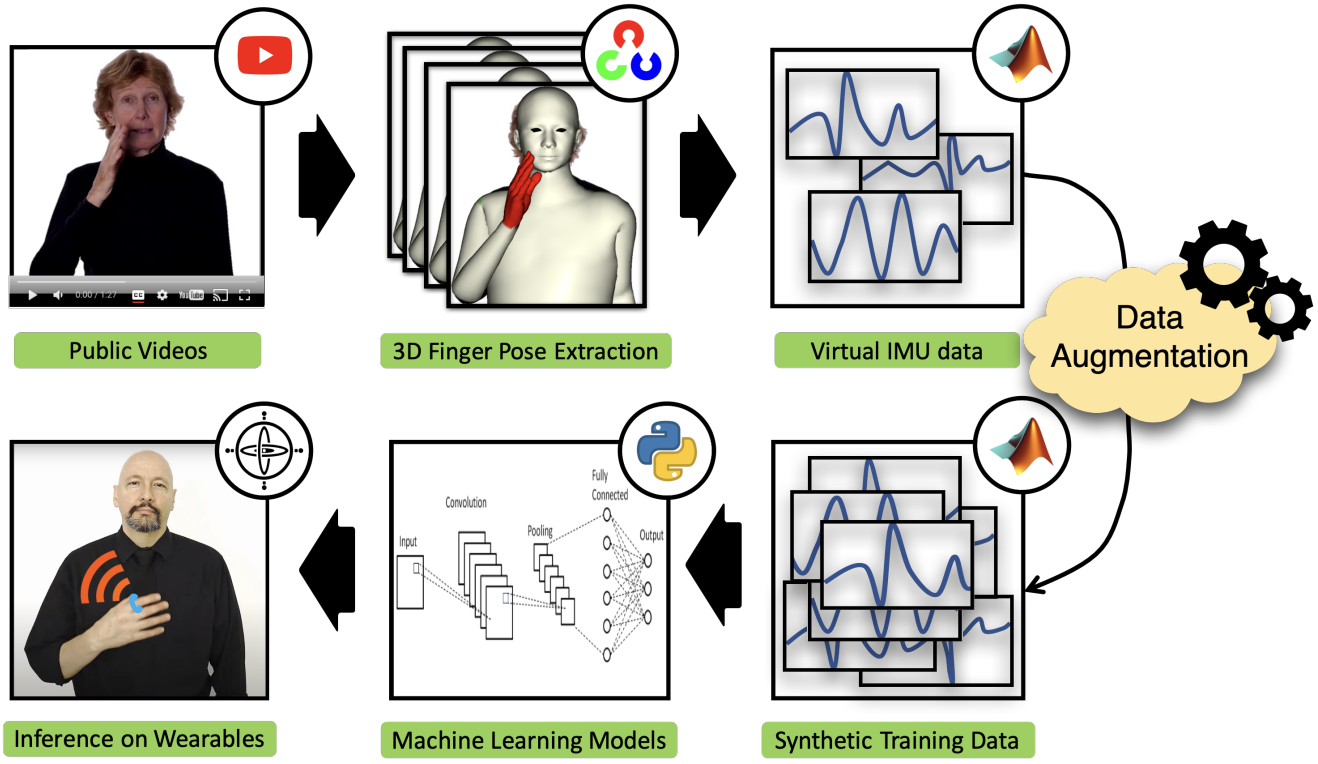


Figure 6.1: The flow of operations in *ZeroNet*. 3D finger pose and locations are first extracted from videos. The location and pose information is transformed into acceleration and orientation that can be captured by inertial sensors. Data augmentation techniques are then introduced to create robust synthetic training datasets. The ML models developed on such datasets are generalizable and directly used for inferences on wearable devices (smart-ring worn on finger) without any training overhead.

Harvesting training data from videos for performing inferences on IMU is challenging because: (i) The IMU and camera have differences in sensing modalities, coordinate systems etc., thus requiring careful pre-processing to transfer knowledge between the two domains. (ii) The speed/orientation of gesturing, and body sizes can differ across users. Similarly, the sensor wearing position and orientation can vary due to natural errors in sensor placement. (iii) The distribution of training data and test data will not match since they come from different sources. Appropriate techniques are needed to generalize the model developed from video-based training data to perform accurate inferences on wearable devices.

In solving the above challenges, *ZeroNet* exploits a number of opportunities. (i) The sensor and camera coordinates can be appropriately aligned by measuring the orientation of the wearable device to perform coordinate transformations. (ii) *ZeroNet* approximates IMU-like sensor data from location estimates extracted from videos by performing systematic finite

differences of locations to derive accelerometer data. Similarly, the angle between finger joints and the vertical plane is extracted from videos to approximate a dimension of the orientation data. (iii) Towards handling body size diversity, the location data from each video is normalized to a measurement corresponding to a uniform body size (for example, by scaling the data by the ratio of the shoulder length of the person in video to a standard shoulder length). (iv) Towards enhancing the robustness and generalizability of ML models, we augment the training data by creating synthetic variants of the data with varying speeds and magnitudes of acceleration. In addition, variants of data with minor shifts in rotations is also added to provide robustness to varying finger orientations or sensor positioning/displacement.

We implement *ZeroNet* on a wearable platform of a button shaped off-the-shelf Mbient Sensor [237] worn as a ring on fingers. We extract training data for 50 gestures of finger motion from a popular public video source of American Sign Language (ASL) tutorial [238]. We develop a CNN based model using this data by exploiting the above enumerated opportunities. Testing results on 10 users achieves a top-1 accuracy of 82.4% and a top-3 accuracy if 94.8% which demonstrates the feasibility of our system. An implementation on Samsung Galaxy S20 smartphones using TensorflowLite validates the low latency and energy efficiency of the system.

A summary of our contributions in the paper include:

(i) Showing the feasibility of harvesting training data from videos for performing inferences on IMU for finger gesture tracking.

(ii) A systematic pipeline that fuses data processing and data augmentation techniques for better generalizability of ML models

(iii) Evaluation with 10 users that shows a top-1 accuracy of 82.4% *and a top-3 accuracy of* 94.8% *over 50 gestures.*

The rest of the paper will expand upon this idea. Sec. 6.2 provides a background on the nature of data in the domains of videos and IMU, while Sec. 6.3 introduces the IMU platform . Sec. 6.4 will design a signal processing pipeline for systematically transforming video-based training data into IMU-based data. Sec. 6.5 will discuss data augmentation techniques to handle the domain difference between training and test data as well as for creating robust model that is generalizable to any new user. Sec. 6.6 will provide results from our experiments. Sec. 6.7 will survey related research and finally we conclude with limitations and future directions in Sec. 6.8.

6.2 Background

The success of human activity recognition in machine learning depends on the availability of large scale annotated datasets. For example *Human 3.6m* [239] has 3.6 million images of various activities such as eating, walking, discussing, sitting, providing-directions, talking on phone, making purchases etc. Similarly, the popular *ImageNet* database consists of 14 million images. In contrast, for wearables, *Daphnet* [240] gait dataset has 5 hours of walking data from 10 subjects and *PAMP2* dataset [241] has 7.5 hours of sensor data from 9 subjects. Such datasets are very small in comparison to vision datasets. Moreover to the best of our knowledge, such datasets do not exist for finger motion tracking that use the recently emerging platform of smart-rings. Towards overcoming this gap, this section briefly discusses extracting finger locations from video data for harvesting training data. We also discuss the nature of IMU data to be approximated with video data.

6.2.1 Video Data

Large amounts of video datasets are publicly available. For example, there are several YouTube videos of sports activities, movie clips of human activities, sign language news etc. Exploiting such datasets for harvesting training data can significantly reduce the overhead of training data generation on wearable devices. In this paper, we harvest training data from a popular public tutorial of sign language gestures [238]. We show the feasibility of recognition of 50 most popular finger gestures without any training data from IMU.

We exploit state-of-the art computer vision techniques for extracting motion data from the videos for training ML models. Fig. 6.1 shows an example of a frame from our video dataset. We exploit techniques in [235] that can extract finger joint locations from simple RGB images, also shown in Fig. 6.1. In particular, Xiang et al [235] use an efficient representation called 3D part orientation fields (POF) to encode 3D orientation of all body parts in a 2D image space. The POFs are learnt by a CNN trained over a large dataset thus learning to predict 3D deformable mesh model of the whole body, face, and fingers. While RGB images do not contain depth information, the CNN model exploits the known priors of shape and pose models of human body in addition to applying constraints of temporal smoothness for extracting 3D motion information. As shown in Fig. 6.1, the whole body shape is extracted from which we only identify the finger locations from the red highlighted region. The extracted finger locations is used by *ZeroNet* to develop ML models for IMU data as elaborated in further sections.

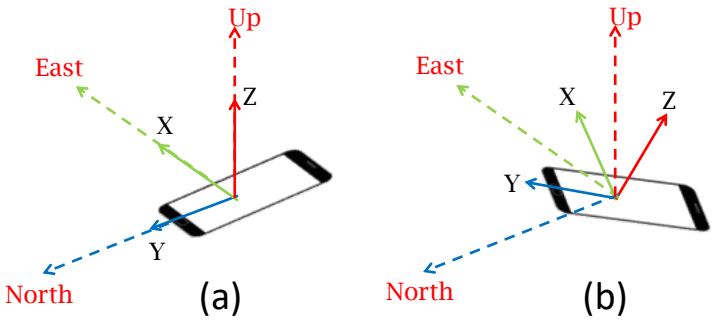


Figure 6.2: (a) Perfectly aligned *local* and *global frames* (b) Misalignment between *local* and *global frames*. Orientation captures the misalignment between *local* and *global frames*

6.2.2 IMU Sensor Data

Inertial Measurement Units (IMU) consists of accelerometer, gyroscope, and magnetometer sensors widely embedded in wearable IoT devices for enabling a number of applications in gesture recognition, augmented reality, smart health etc. We provide a brief overview of the 3D orientation of an object since it plays a critical role in modeling the data captured by these sensors.

Consider a *global frame* of reference pointed towards "Up", "East", and "North" directions (Fig. 6.2). Consider an object (e.g., IMU sensor) whose *local frame* of reference is also shown in the figure. While the two frames are perfectly aligned in Fig. 6.2(a), there is a misalignment between the two frames as shown in Fig. 6.2(b). The *3D orientation* of an object captures this misalignment between the *local* and *global frames* of reference. Consider a vector V whose representation in the *local* and *global frames* are $V_l = [X_l\ Y_l\ Z_l]$, and $V_g = [X_g\ Y_g\ Z_g]$ respectively. The 3D orientation of the object can be mathematically quantified using a 3×3 rotation matrix R which rotates the vector between the two frames of reference as indicated below.

$$\begin{bmatrix} X_l & Y_l & Z_l \end{bmatrix} R = \begin{bmatrix} X_g & Y_g & Z_g \end{bmatrix}$$

When an accelerometer sensor is under rest, it measures the projection of the gravity vector on its three axes [242]. Similarly, the magnetometer sensor measures the projection of the earth's magnetic field on its three axes. Since the acceleration due to gravity and the geomagnetic field are globally known vectors, the local measurements of these values using the sensors can ideally be used for computing the rotation matrix R described above to quantify the orientation of an object. However, in reality, the mobility of the sensor can corrupt the measurements of gravity by the accelerometer, as well as the electromagnetic

Figure 6.3: Button sized IMU worn as a ring

interference can interfere with the magnetometer. Therefore, the gyroscope sensor data which measures the change in orientation (angular velocity) can be fused with estimates of orientation from accelerometer and magnetometer to compute accurate 3D orientation estimates of an object [136].

An accelerometer sensor measures the superposition of the gravity and acceleration due to the linear motion of the wearable device. The measurement is relative to the sensor's *local frame* of reference. Therefore, the orientation estimates as discussed above is useful not only in converting the accelerometer measurements to the *global frame* of reference but also in subtracting the component of gravity from the acceleration measurements.

6.3 Platform Description

We begin with a simple platform with a ring-like sensor worn on the index finger as shown in Fig. 6.3. Note that all fingers are involved in gesturing, but we place the sensor only on the index finger. While we believe this is sufficient to show the feasibility of harvesting training data from videos, this will cause miss-classifications among gestures with similar motion of the index finger and different motion of other fingers. However, surprisingly, the accuracy with just index-finger data is significant with very few miss-classifications due to the specific reason noted here (details in Sec. 6.6). While the miss-classification rate might increase with number of classes, we discuss opportunities with additional techniques and sensors in Sec. 6.8. The majority of the study places the sensor on the index finger since it is more frequently involved in gestures in our video dataset. However, we also conduct experiments to understand the best

placement option among other fingers (Sec. 6.6).

Smart rings that can pair with phones wirelessly to stream information as well a monitor activity are already available on the market [127, 128]. For example, the Oura ring [128] is popular as a sleep tracking device and weighs between 4 and 6 grams, which is even lighter than conventional rings, and packaged in a stylish design. It is low intrusive with users finding it comfortable for wearing day and night, gym, pool etc [129], thus receiving favorable online reviews for usability [129–131]. However, most of these platforms are closed and do not provide access to raw sensor data. Therefore we use a button-shaped sensor from MbientLabs [237] snugly fit on the finger like a ring as shown in Fig. 6.3. The sensor streams data wirelessly to a smartphone which runs the ML models for gesture recognition. The ring generates 9 axis IMU data - 3 axes each for Accelerometer, Magnetometer, and Gyroscope. This forms the input to *ZeroNet*.

6.4 Synthetic Training Data from Videos

The 3D locations captured from the video will be transformed into synthetic accelerometer and orientation data for training the ML models. A natural first step would be to simply double differentiate the index finger location as extracted from the video to obtain the acceleration of the index finger. However, such a simple differentiation will not emulate the accelerometer data because of a number of differences between IMU and video data. In this section, we elaborate these differences together with approaches in *ZeroNet* to address these differences. We begin by discussing the basic pre-processing steps.

6.4.1 Pre-processing

A number of simple but critical pre-processing steps are needed to match the distribution of the video and IMU dataset. We enumerate the main steps here: (i) A low pass filtering with a cutoff frequency of 10Hz was applied on both video derived acceleration and IMU acceleration. (ii) The orientation data extracted from videos posses a characteristic shape mainly because of the noise in the camera data. Simply using these orientation estimates made the CNN model memorize the shape and overfit. Thus, we regularized the orientation data using a smooth, low parametric function so as to prevent the CNN model from memorizing the noise in the data.

6.4.2 Extraction of Acceleration

Coordinate differences: The location data captured by cameras is relative to the camera's frame of reference. However, the locations can be transformed into torso coordinate frame (TCF) as shown in Fig. 6.4. We chose our x-axis as the line joining the two ends of the shoulder

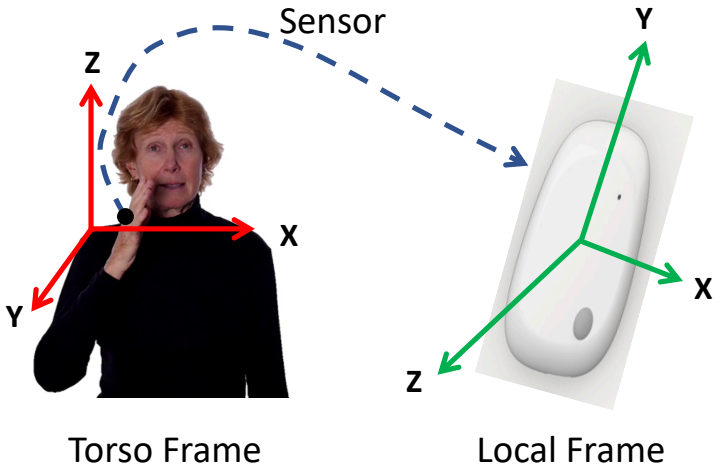


Figure 6.4: The camera's motion data in *Torso Coordinate Frame* can be aligned with the sensor measure data relative to *Local Frame* using orientation estimates of sensor

when the user is in a *stable pose*. Similarly, we chose the z-axis to be in the plane of the torso but perpendicular to x-axis. The y-axis is perpendicular to these two axes. Since we extract entire shape of the human body using the work in [235], we identify the appropriate shoulder and torso joints corresponding to the TCF. We then project the extracted locations from the camera into TCF.

On the other hand, the acceleration measured by the sensors will be in the *local frame* of reference which depends on the instantaneous orientation of the sensor as depicted in Fig. 6.4. Therefore, *ZeroNet* first converts the acceleration into the *global frame* of reference. The difference between the *global frame* and the user's facing direction can be roughly computed when the sensor is in vertical free-fall position or if the user is walking a few steps [243, 244]. We adopt this approach in this paper for computing the difference between TCF and *global frame*. Thus, the acceleration is first converted to *global frame*, and then to TCF by using the orientation estimates of the sensor. After this transformation, the acceleration due to gravity is subtracted from the result since the accelerometer measurement includes the sum of gravity and linear acceleration. The video and IMU data will now be comparable with each other.

Fig. 6.5(a) compares the z-axis accelerometer data with double differentiated data of video

locations before such coordinate transformations for a hand gesture. Evidently, the two data look dissimilar. On the other hand, after performing appropriate transformations, the two sources of data look similar as depicted in Fig. 6.5(b), which indicates the z-axis acceleration along TCF.

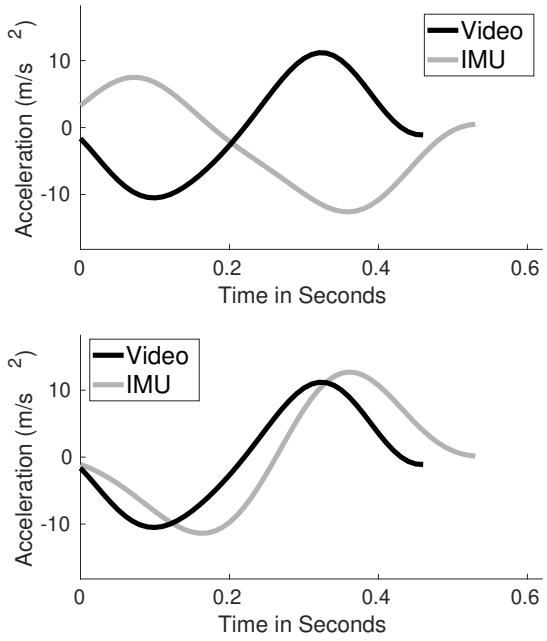


Figure 6.5: (a) The acceleration data from camera and video do not match before coordinate alignment between *TCF* and *LF* (b) The data from the two domains match well after coordinate alignment between *TCF* and *LF*

Double differences: While we considered tools like IMUSim [245] to convert location data from videos to IMU data (e.g. acceleration), there is no support for simulating finger joints. Therefore, we perform finite double differences as indicated by the equation below, as also explored in prior work [103, 104].

$$a_t = \frac{p_{t-\Delta t} + p_{t+\Delta t} - 2 \cdot p_t}{\Delta t^2}$$

This extracts accelerometer data a_t from locations p_t extracted from videos. While the IMU provides instantaneous acceleration, the finite time double differences is only an approximation. Choosing a smaller Δt reduces the error in approximation due to finite differences. However, smaller Δt also decreases the signal to noise ratio (SNR) of the generated acceleration signal because the change in location may be too small over a small time interval whereas the noise in the data is independent of time. We choose a value of Δt as 0.1s which provides a tradeoff

that works well in practice. An example is depicted in Fig. 6.5(b) where finite differences are performed after the preprocessing steps such as low pass filtering.

Body size normalization and camera parameters: Difference in body sizes of users can create differences in the recorded sensor data even for the same gesture. In addition, the primary unit of estimate of locations from images is in pixels. Extraction of location in units of cms from public videos will need information or estimates about the camera parameters [31]. Towards handling body size differences as well as to eliminate the need for camera parameters, we normalize all location estimates from camera to the size of a standard human. In particular, we measure the shoulder length in pixels and scale it with factor such that the shoulder length is 27 cm. Such scaled locations are used for deriving synthetic accelerometer data. During testing, the accelerometer measurements from a human are similarly scaled depending on how their shoulder length compares with the standard length (27 cm). Fig. 6.6 shows an example of comparison between video and IMU data before and after normalization. Experimentally validated, the normalization step enables better similarity in sensor data despite the difference in body sizes of users and not having the camera parameters.

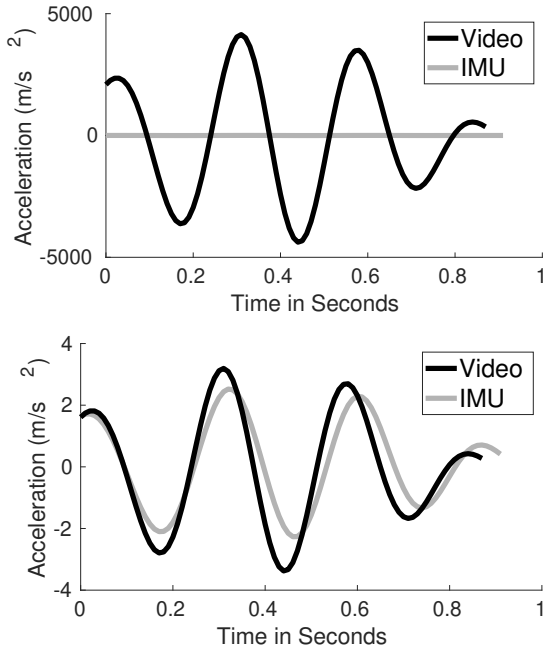


Figure 6.6: (a) The data from video and IMU domains can vary widely in magnitude because of differences in body sizes and units of measurements (b) Normalization techniques in *ZeroNet* renders the data from the two domains comparable

6.4.3 Extraction of Finger Orientation

Fig. 6.7 shows the metacarpophalangeal (MCP) and proximal interphalangeal (PIP) joints of the index finger. The angle made by the line joining these two joints with the vertical plane can be extracted from these videos. The same piece of information can be extracted from the orientation estimates of the IMU as indicated in the below equations.

$$y_{proj,xz} = R_f * R \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} - \begin{bmatrix} 0 & 1 & 0 \end{bmatrix} R_f * R \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}$$

$$angle = \arccos \frac{y_{proj,xz}^T \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}}{|y_{proj,xz}|}$$

Here, $y_{proj,xz}$ denotes the projection of the direction of the finger (line joining MCP and PIP joints) on the *XZ* plane. The sensor is roughly aligned such that its local y-axis is along the direction of the finger, but no careful calibration is needed. R is the 3×3 rotation matrix, R_f indicates the misalignmentment between the user's facing direction and the magnetic north. We compute this by adopting ideas from past work [243, 244]. Thus, the $angle$ between MCP-PIP joints and the vertical axis as indicated above will be used as a virtual orientation data for training the ML models in *ZeroNet*. While the orientation estimates of a IMU sensor is 3 dimensional, we restrict ourselves to extracting the 1 dimensional $angle$ information as discussed above mainly because: (i) We can extract it reliably and compares well with the same information extracted from IMU. (ii) We found that estimating rotation along the axis of the index finger although possible in theory from the information extracted from videos, proved to be unreliable and erroneous in practice.

6.5 Gesture Recognition Models with Synthetic Training Data

We explore two methods for exploiting the training data extracted from videos for performing gesture recognition on IMU: (i) A simple DTW based model (ii) A Convolutional Neural Network based machine learning model

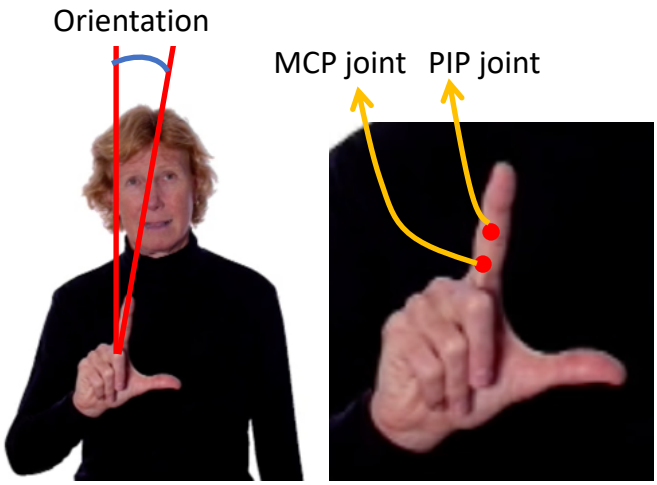


Figure 6.7: The angle depicted here can be extracted from videos and used as a training data for inferences on IMU

6.5.1 Dynamic Time Warping

We begin by using dynamic time warping (DTW) [134] to compare the IMU data from an unknown user gesture to video-based training dataset for gesture recognition. Briefly, DTW is a pattern matching technique that inspects the overall shape of two signals to determine their similarity. For example, Fig. 6.8(a) shows the z-axis accelerometer data from IMU and the synthetic accelerometer data extracted from a video of the same gesture. Although the overall shape is similar, parts of the motion traces happen at a faster rate or earlier for IMU while other parts happen slower. DTW uses a dynamic programming optimization to minimally compress and stretch the two sequences relative to each other such that they provide the best overlap. Fig. 6.8(b) shows the two sequences after DTW optimization. DTW is known to do a good job of matching such series with similar overall shape. The residual differences between the two series determines the similarity score among them. The similarity score of an unknown gesture is compared with all gestures in the training data. The gesture with the best match would correspond to the correct gesture with high probability. The 3-axis accelerometer data and the orientation of the index finger is used for performing the DTW matching as described above.

6.5.2 Convolutional Neural Networks

Towards increasing the robustness of recognition, we take a data-driven ML approach in addition to DTW. The architecture of the model is depicted in Fig. 6.9. The success of ML

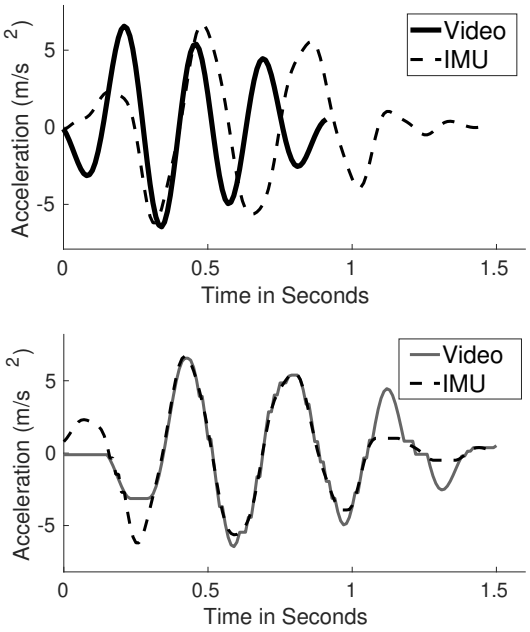


Figure 6.8: (a) Accelerometer data for "More" extracted from video of one user in comparison with IMU data of another user (b) Data from IMU is compressed and stretched to match with video by DTW

models depend on availability of large scale high quality training datasets. In addition to extracting the training data from videos, we exploit the following data augmentation techniques to ensure stability, robustness, and convergence of the above ML model.

DTW based augmentation: The performance of gestures will vary widely across users. The speed of hand motion is one of the metrics that can vary across users. Various parts of the gesture might be performed at a faster or slower pace by different users. Towards making the ML models robust to such variations, we augment the training data by injecting such variations into existing training data. In particular, we stretch and compress different parts of the training data with different factors to create new training data from existing samples.

Fig. 6.10 shows an example where two sequences A and B are aligned using DTW. Fig. 6.10 (b) shows the correspondence between samples in the two sequences, whereas Fig. 6.10 (a) depicts the same in matrix form. Given a training data sample A, we generate random matrices similar to Fig. 6.10 (a) to create dynamically stretched and compressed versions of the training data sample. In creating these matrices, we resample the original time series of the training data with a stochastic non-uniform sampling such that compression/expansion ratio

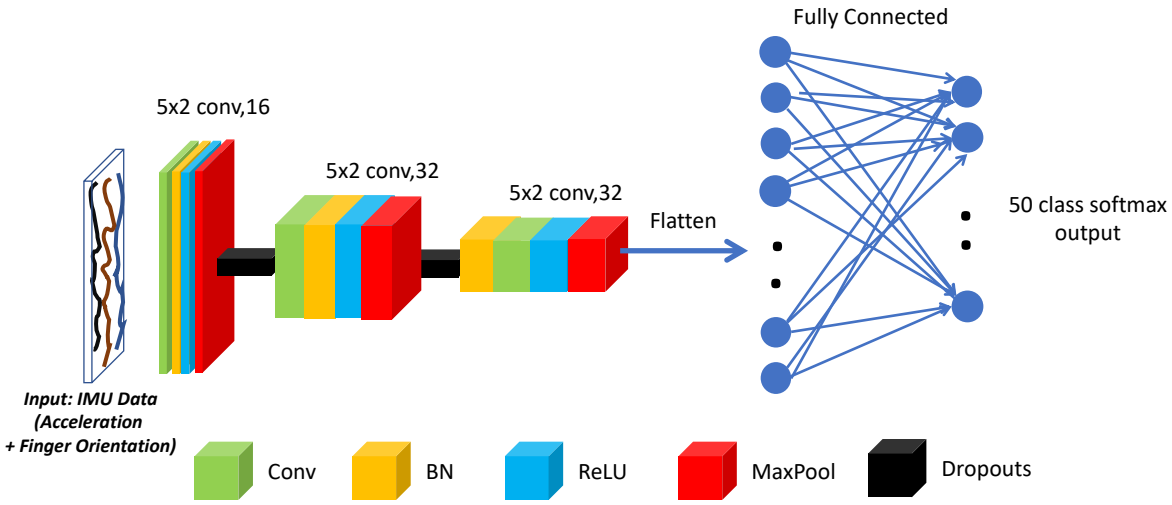


Figure 6.9: The CNN based machine learning model in *ZeroNet*

varies between 0.25 to 2. Appropriate interpolation strategies are used since the resampling positions may not coincide exactly with the positions in the original time series. Fig. 6.11

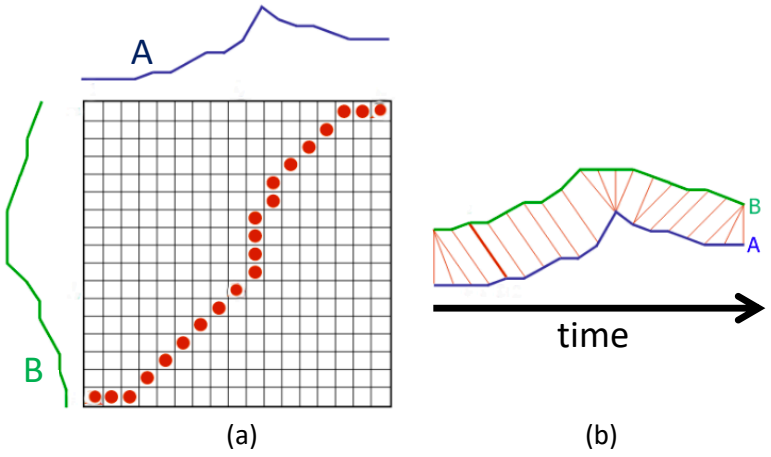


Figure 6.10: DTW alignment matrix between two sequences A and B. Pictures adopted from [9]

shows an example where two variants of new training data has been created from an existing training data.

Orientation Variation: Similar to variations in gesturing where users perform at different speeds, the orientation of the hand can vary during motion. Such variations can also happen because of minor changes in the sensor wearing position or orientation across users. Fig. 6.12

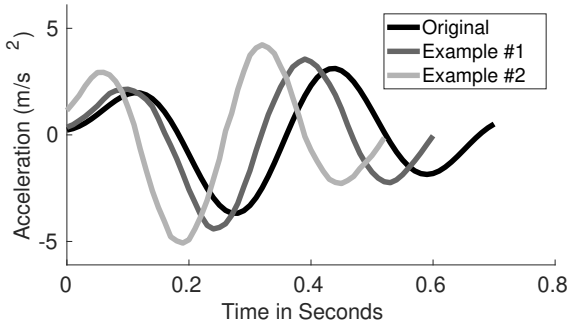


Figure 6.11: DTW synthetic training data

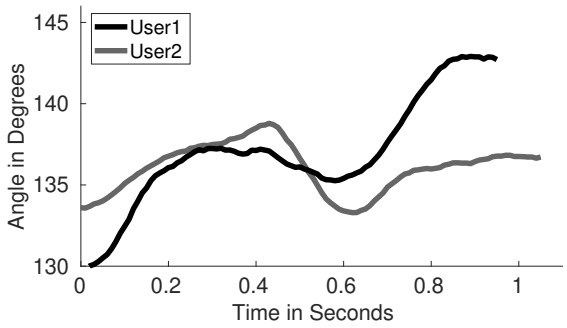


Figure 6.12: Variation in orientation across users

shows an example where the same gesture is performed by two users with a minor shift in the hand orientation. The ML model must be robust for adapting to such natural variations. Therefore we augment training datasets emulating variations in hand orientation while gesturing. The injected variations range from 0 to 10 degrees, but they are not random, rather they ensure smoothness and continuity thus emulating a realistic gesture with small changes in orientation. Fig. 6.13 shows examples of augmented data with varying orientations for a given gesture.

Temporal Clipping: We also hypothesize that the start and end periods of performance

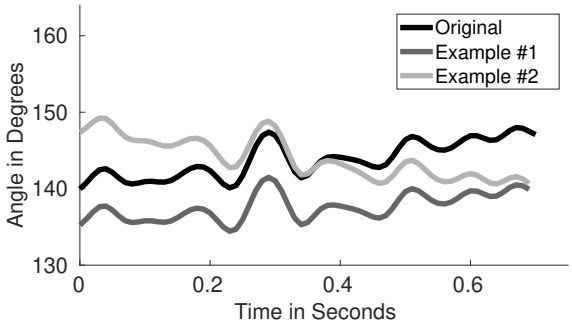


Figure 6.13: Examples of synthetic orientation data

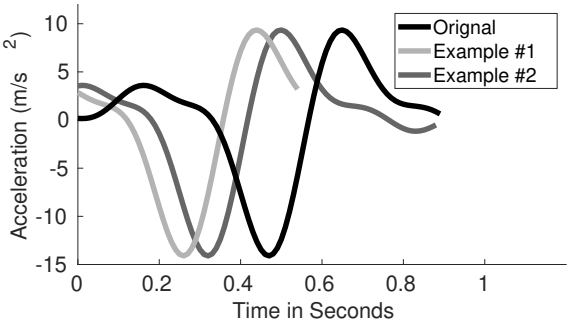


Figure 6.14: Examples of temporal clipping

of gestures by several users will vary. Different users might start the gesture from slightly different positions as well as end the gesture prematurely or continue with extra motions beyond the gesture. To help the model generalize under such diversity, we augment training data by introducing versions of the training data with minor extrapolations or trimming of samples at the begin and end of the gestures. Fig. 6.14 shows an example where two variants of synthetic training data are added with random clipping at the beginning and end of an original training data.

6.6 Implementation and Evaluation

Implementation: The sensor frontend includes an Mbient sensor [237] as described in Sec. 6.3 which is worn on the index finger as a ring. The 9-axis IMU data including accelerometer, gyroscope, and magnetometer data is streamed to a smartphone. *ZeroNet* is implemented on a combination of desktop and smartphone devices. The machine learning architecture is implemented using TensorFlow [215] packages and the training is performed on a desktop with Intel i7-8700K CPU, 16GB RAM memory, and Nvidia GTX 1080 GPU. We use the Adam optimizer [216] with a learning rate of 1e-3, β_1 of 0.9 and β_2 of 0.999. To avoid over-fitting issues that may happen in the training process, we apply the L2 regularization [217] on each CONV layer with a parameter of 0.01 and also add dropouts [200] with a parameter of 0.1 following each RELU activations. Once a model is generated from training, the inference is done entirely on a smartphone device using TensorFlowLite [10] on a Samsung Galaxy S20 smartphone with Android operating system.

User Study: All reported results in the paper are generated from a systematic user study campaign. The study evaluates the classification accuracy of 50 gestures that represent the

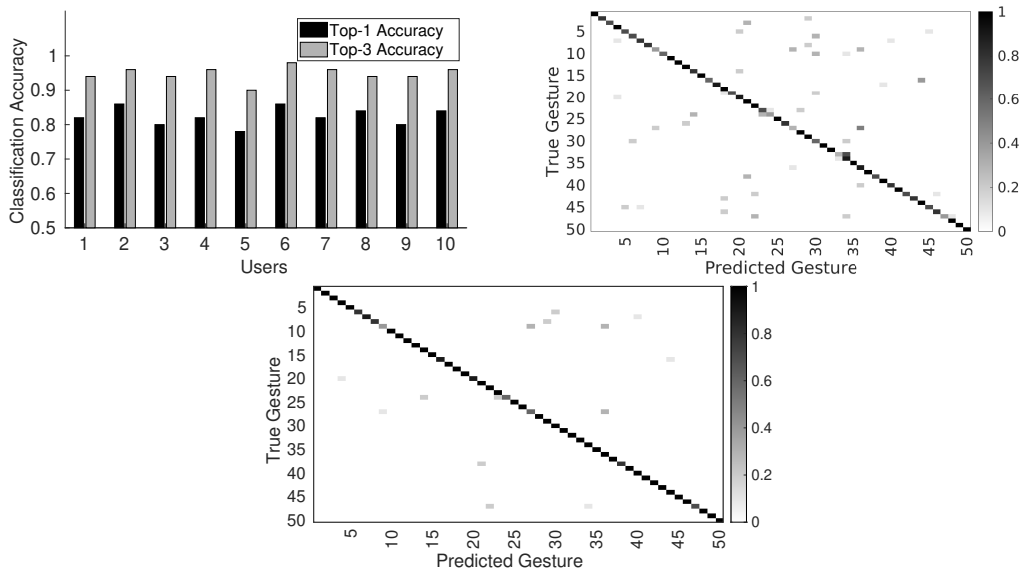


Figure 6.15: (a) Overall Accuracy vs Users (b) Top-1 Accuracy vs Gestures (c) Top-3 Accuracy vs Gestures

top 50 ASL words. The training data is extracted from the following video source [238]. We recruit 10 users aged 21-32 and weighs between 47 to 96kgs. It includes 7 males and 3 females. During the data collection process, the user is first shown the video of a gesture. The user practices performing the gesture several times. When the user feels comfortable performing the gesture correctly, we let the user perform the gesture 5 more times and we record the sensor data during this period. After this process, we repeat the procedure for the next gesture until we finish collection of the data for all 50 gestures. The entire recorded dataset during the study is solely used as a 'test data' since the training data is extracted entirely from videos.

We specifically aim to answer the following questions.

- What is the overall gesture recognition accuracy? *(Figs. 6.15(a), Figs. 6.20)*
- Is the accuracy consistent across diverse gestures? *(Figs. 6.15(b), Figs. 6.15(c))*
- How does the accuracy vary across users? *(Figs. 6.15)*
- In cases of errors in recognition, what is the rank of the correct gesture among all the 50 gestures? *(Figs. 6.16)*
- How does the accuracy vary with the speed of gesturing? *(Figs. 6.17)*
- How does the accuracy vary with sensor placement on the hand? *(Figs. 6.18)*

- What is the accuracy of the model transferred to the left hand? *(Figs. 6.19)*
- What is the role of various techniques of data augmentation in the final accuracy metric? *(Figs. 6.21)*
- How does the accuracy vary with the size of the synthetic dataset? *(Figs. 6.22)*
- How does *ZeroNet* compare with models fine-tuned with real-data or models fully trained on real data? *(Figs. 6.22, Figs. 6.23)*
- What is energy, latency, and compute profile of executing the ML models on embedded devices? *(Figs. 6.24)*

Robustness to sensor wearing positions and diverse gesture patterns: Fig. 6.15(a) depicts the overall accuracy as a function of users. Evidently, the accuracy is stable across users, body sizes, motion patterns etc. In addition, the sensors were mounted naturally on the fingers with y-axis roughly along the direction of the index finger. There was no special calibration and hence the position/orientation across users would naturally vary. However, the accuracy is robust to such variations. This is because of the inbuilt robustness to such natural variations through the data augmentation techniques incorporated in the design of *ZeroNet*. While the top-1 accuracy is 82.4%, the top-3 accuracy is around 94.8% which indicates promise for future improvements.

Accuracy over gestures: Fig. 6.15(b) shows the confusion matrix over all 50 gestures in our dictionary. The performance is consistent across all gestures. However, in certain special cases, such as the gesture for "mother", and "father", there can be miss-classifications because the index finger motion for these two gestures are very similar. Fig. 6.15(c) shows the confusion matrix for top-3 accuracy which shows a higher accuracy because in many cases of miss-classifications the correct word is occupies the second or third place in the rank of softmax probabilities.

Rank of incorrect gestures: We provide further breakup of cases where the top identified gesture is incorrect. Fig. 6.16 shows the rank of the correct gesture in case of erroneous detections. Evidently, majority of the cases are rank-2 and 70.5%, 83.0% of cases are in top-3 and top-5 ranks respectively. This indicates that appropriate application specific prior information or context can be exploited to further imporve the accuracy of *ZeroNet*.

Accuracy over speed: Fig. 6.17 provides a breakup of accuracy of gestures executed at varying speeds. Note that in addition to some gestures being inherently slow or fast paced,

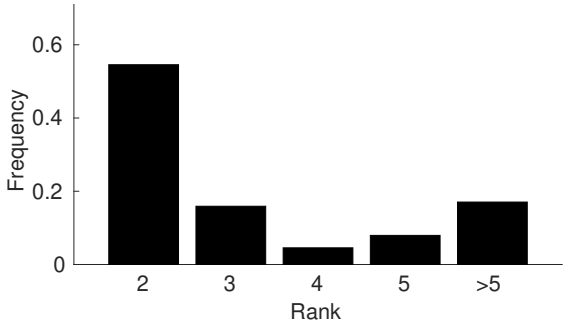


Figure 6.16: Rank of correct gestures for erroneous cases.

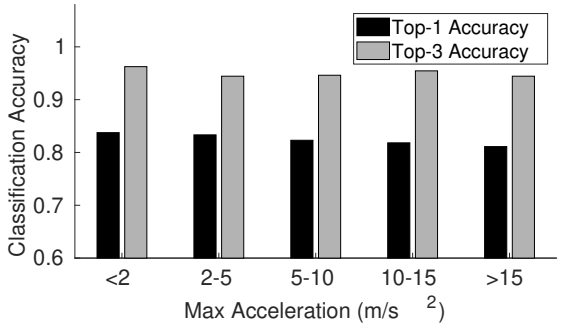


Figure 6.17: Accuracy over speed of gesturing

variations in pace can also occur because of user diversity. Regardless of the reason of variation, the accuracy is robust at various possible speeds.

Accuracy vs Finger Position: An advantage of harvesting training data from videos is that optimal sensor placement can be determined for any given application where there is a tradeoff in number of sensors that can be used due to reasons including usability, accuracy, power consumption etc. We conduct a small study to determine the optimal sensor placement among index, middle, and ring fingers for the top 20 gestures from our video dataset [238]. The little finger and thumb were excluded in the study since it is not comfortable to wear sensors on those fingers. Fig. 6.18 shows that the top-1 accuracy values are 93.4%, 88.3%, and 85.1% for index, middle, and ring fingers respectively. This indicates that the optimal sensor placement among the three fingers is the index finger for the set of gestures considered in this application.

Model transfer between right and left hands: Fig. 6.19 shows the accuracy when the left hand was used in gesturing. This is useful when the training data from videos of right-handed users is used for performing inferences on left-handed users. The training data captured from the right hand was appropriately mirrored to emulate a training data for the left hand. This includes making the x-axis in Fig. 6.4 negative and projecting the acceleration and orientation

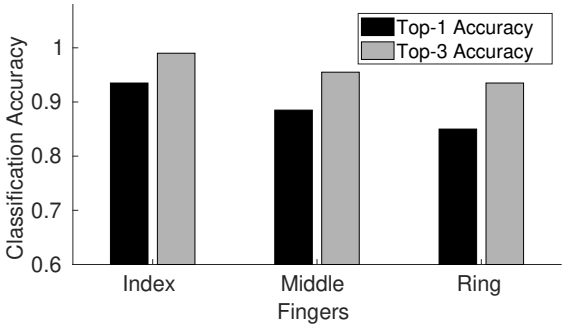


Figure 6.18: *ZeroNet* can generate training data for any finger position, thus facilitating optimal sensor positioning

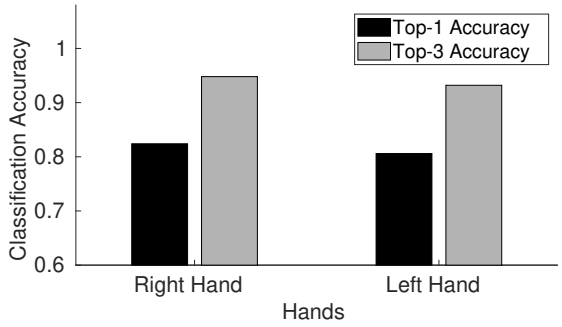


Figure 6.19: Model transfer from right to left hand

to the new TCF relative to the left hand. The transformed training data was used to train the ML model in Fig. 6.9 to perform inferences when the sensor is worn on the left hand. Evidently, the accuracy for such inferences is same as the right hand.

Performance comparison across techniques: Fig. 6.20 provides a breakup of accuracy across techniques. Basic DTW already achieves a reasonable accuracy of 59.4%. On the otherhand, the accuracy of the basic CNN model is slightly lower than DTW because of the inability to generalize to diversity in user motion patterns. However, data augmentation techniques in *ZeroNet* can make the CNN model robust to speed of gesturing, sensor positions, orientation variation, noise etc, thus boosting the accuracy to 82.4%.

Breakup of performance gain from data augmentation: DTW based augmentation, rotation based augmentation and time clipping individually achieve accuracies of 70.7%, 53.4%, and 60.8% respectively as shown in Fig. 6.21. DTW-based augmentation performs the best while the other techniques also offer non-trivial gain in performance relative to a baseline without data augmentation. However, combining all of them yields the best performance.

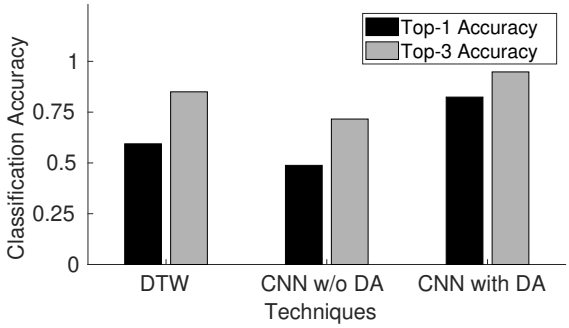


Figure 6.20: Performance comparison across: (i) DTW (ii) CNN (iii) CNN with data augmentation (DA)

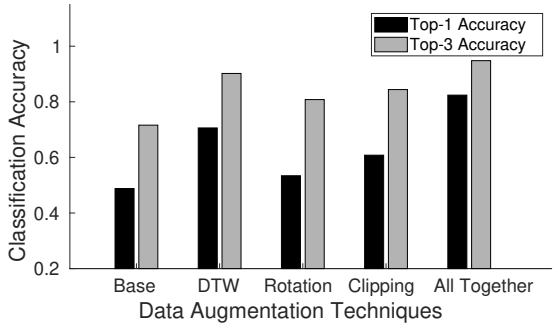


Figure 6.21: Role of individual data augmentation techniques

Training with Synthetic vs Real Data The first bar in Fig. 6.22 shows the performance accuracy of training with real data alone. Evidently, the small size of the data leads to overfitting and poor generalization thus leading to overall low accuracy. On the other hand, the last bar depicts the effect of training with synthetic data which together with data augmentation techniques leads to better generalization of the ML models and higher accuracy.

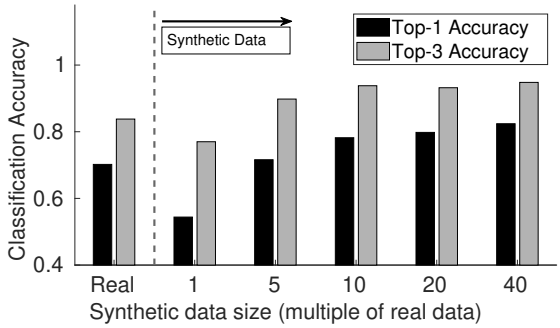


Figure 6.22: Diversity in synthetic data leads to better generalization of the CNN model

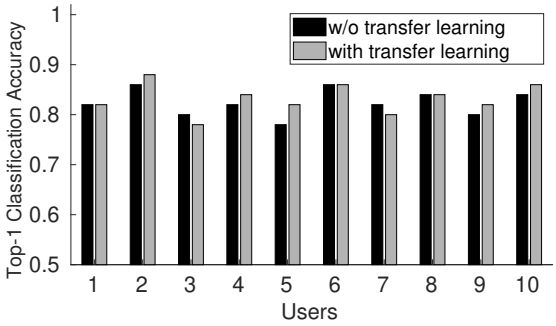


Figure 6.23: Model fine tuning with real IMU data

Effect of the size of synthetic data: Fig. 6.22 depicts the performance of the CNN model as a function of the size of the synthetic data. The x-axis label denotes the size of the synthetic data in multiples of the size of the real data. Evidently, higher size of synthetic data creates more robustness in the training examples that the ML model sees during training. Thus, the overall accuracy improves with size, ultimately saturating when the size of the synthetic data is 40 times of the real data.

ML models fine tuned with real IMU data: The CNN model that was trained in *ZeroNet* using synthetic IMU data from videos was fine-tuned [209, 210] with real IMU data. Fig. 6.23 depicts the performance over users. Leave-one-out cross-validation was adopted across users. The fine tuning improves the performance only marginally. We believe this is because the data augmentation techniques sufficiently cover the space of variations thus generalizing the CNN model to the maximal extent.

Energy, latency, and compute: we use Batterystats and Battery Historian [220] toolkits for profiling the energy of the TensorflowLite model for inference using CNN and the DTW-based classification model. We compare the difference between energy consumption in two states (i) When the device is idle with screen on. (ii) The device is making inferences at a rate of 2 gestures per second. The idle display-screen on discharge rate 4.95% per hour while the discharge rates for various techniques is depicted in Fig. 6.24. Evidently, the power consumption profile of the CNN model is very low. We believe the CNN model is more efficient than the simple DTW because of inbuilt optimizations within the TensorFlowLite library [10]. The latency results are very much correlated with power consumption results. In particular, each classification takes $2.2ms$ on average with CNN whereas it takes $266.4ms$ with the DTW model. We believe the overall power consumption and latency profiles of the CNN model enables energy efficient real time performance.

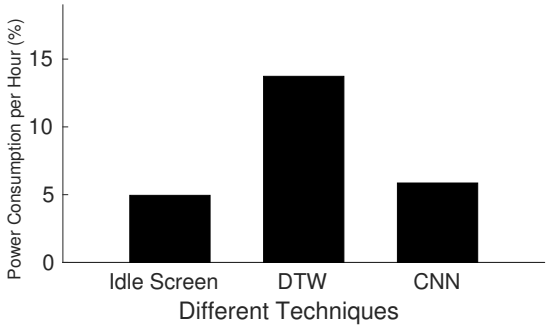


Figure 6.24: The power consumption profile the CNN model is better than simple DTW because of builtin optimizations in TensorFlowLite [10]

6.7 Related Work

Inertial Sensors: Inertial sensors have been used in many localization and gesture tracking applications. UnLoc [63] fuses information from smartphone sensors for extracting characteristic fingerprints in indoor environments for localization. RisQ [64] recognizes smoking gestures for appropriate intervention measures using smartwatches. Similarly, smartwatches are used for eating activity recognition [65] and measuring calorie intakes. Smart rings are also being used for ASL gesture recognition in recent times [224, 246]. DUI [66] detects blood alcohol level based on user performance on smartphone activities. Other applications have been explored in the areas of augmented and virtual reality, sports analytics, smart-health, and security [67–70]. In contrast to these works that create training datasets with user studies, crowdsensing etc, *ZeroNet* exploits harvesting training data from publicly available videos.

Vision: Depth cameras including kinect [29] and leap motion [30] sensors have revolutionized the gaming industry by gesture interfaces. Use of depth camera is one way to capture finger motion. However, advances in machine learning, availability of large training datasets as well as techniques for creation of synthetic datasets have enabled precise tracking of finger motion even from monocular videos that do not contain depth information [31, 32, 150]. While such works are truly transformative in nature, we believe wearable based solutions have benefits over vision based approaches which are susceptible to occlusions, lighting, and resolution. In addition, wearable devices offer ubiquitous solution with continuous tracking without the need of an externally mounted camera.

Radio Frequency (RF): RF including WiFi, RFID, and mmWave hardware have been used for a number of human activity recognition applications. WiSee [51] can detect hand

gestures by measuring doppler shifts from WiFi reflections. 3D pose of the human body has been detected even behind occlusions such as Walls using wireless body reflections [56, 57]. Heart rate, breathing, and physiological signals of interest to healthcare applications have been detected using RF signals [58, 59]. Google project Soli [60] can detect fine grained finger gestures using mmWave reflections. While RF based tracking, like vision, is completely passive, we believe the advantage of wearable device is being completely ubiquitous without the need for any external infrastructure.

Transfer Learning from Videos: Deep Inertial Poser [100] uses synthetic data from motion capture videos (from cameras like ViCON [101]) instead of public videos for training human pose tracking algorithms with 6 on-body IMUs. Such motion capture cameras can provide high quality training data with mm level accuracy. However, creating such datasets requires 6-8 costly ViCON cameras. We believe using publicly available videos is an easier alternative. More recently, several innovative works [102–106] have explored the use of YouTube-like videos for training human activity recognition (HAR) on wearable sensors. In contrast to such works that classify tens of large scale motion activities (running, sitting, eating etc.), *ZeroNet* performs recognition of fine grained finger motions over a larger class of gestures with potential to applications in augmented and virtual reality, sign language recognition etc. In addition, *ZeroNet* fuses the harvested training data with data-augmentation techniques for better robustness of ML models.

Data Augmentation: Data augmentation enriches the quality of datasets to help ML models generalize well and exhibit higher accuracy and robustness with limited quantity of training data. Transformation such as rotation, scaling, translation and elastic distortions on images have been explored to create more training data from existing datasets. [90–93]. Similarly, image cropping, flipping, color shifting, and whitening are other techniques to create new training data from existing datasets [94]. In the area of automatic speech recognition (ASR), data augmentation techniques such as frequency axis distortions [95], speech rate variations, vocal tract normalization [96] etc have been explored to improve the accuracy. In a similar spirit, *ZeroNet* incorporates ideas in data augmentation for IMU datasets for better accuracy, robustness, and generalizability of ML models. This is particularly important in the context of IMU data since there is no large scale public datasets like computer vision or speech. Data augmentation techniques have been explored in the context of wearable sensing for parkinson disease gait monitoring [97] and construction activity monitoring [98]. More recently, data augmentation for human activity recognition has been extensively studied in [99] for several benefits including robustness to sensor wearing positions.

Machine learning algorithm: Transfer-learning based domain adaptation is popular in vision and speech. For example, AlexNet model [94] pretrained on ImageNet database [204] has been fine-tuned for classifying images in medical domain [205], remote-sensing [206] and breast-cancer [207]. Similarly, a pre-trained BERT language model [208] has been fine-tuned for tasks in text-summarizing [209], question answering [210] etc. Adversarial domain adaptation [225] using generative adversarial networks (GAN) is popular. Here, an unsupervised game theoretic strategy is used to transform the distribution of the feature representations from the target-domain into the distribution of the source-domain on which the model was trained. If successful, the model trained on the source domain is directly useful for performing inferences on a target domain. Similarly, other architectures for learning feature transformations to adapt the feature representations from a source domain to a target domain have been proposed [226]. Other works introduce advancements in machine learning for resource-constrained devices [247–250]. However, such techniques are hard to apply to our problem domain since this still requires enough real training data (atleast in unlabelled form) from IMU to achieve sufficient convergence of the domain adaptation process. Furthermore, each user's finger motion pattern as well as natural variations in sensor wearing positions could lead to different distributions in the sensor data [99, 234] thus entailing more real training data under each setting. On the other hand, *ZeroNet* performs comparable to semi-supervised domain adaptation techniques [205, 206] which need partial labelled real IMU data and even outperforms models fully trained on our own real IMU dataset. We believe *ZeroNet*'s ability to provide promising accuracy without any training overhead is a first step towards generating data for unsupervised domain adaptation. Our approach is related to zero-shot learning [251], where a ML model is trained to predict classes for which no training examples has been observed. Appropriate representations are learnt for both training examples and class labels. By learning the mapping between representations of known training examples and their classes, the mapping between representations of a new example is made even if it belongs to an unseen class. One difference between *ZeroNet* and classical zero-shot learning is that zero-shot learning needs training data from the target domain for some classes, whereas *ZeroNet* does not need training data for any classes.

6.8 Discussion and Future Work

Exploiting large scale video datasets: *ZeroNet* only scratches the surface in harvesting training data from videos. 300 hours of videos are uploaded to YouTube every minute for human activities ranging from sports, tutorials, physical exercises, speech, daily activities

(cooking, eating, jogging) etc. Exploiting more videos for building ML models can enhance the robustness.

Automated data augmentation: In *ZeroNet*, the amount of perturbations introduced in the data for augmentation is fixed. Automated data augmentation [252] is an active area of research where the parameters for data augmentation can be modeled as a learning problem. We plan to incorporate the innovations from this area into *ZeroNet* as a part of the future work.

Augmented and Virtual Reality applications: AR and VR applications benefit from fine grained tracking of hand and finger locations. Towards pushing the limits of accuracy, *ZeroNet* will exploit video-based training data for free form tracking of 3D finger joint locations. Similar to our analysis on finding the optimal finger to place the sensor, enough training data can be generated from videos for analyzing the tradeoff between number and position of placement of sensors and the expected accuracy.

6.9 Conclusion

Application of ML models for finger gesture recognition can enable a number of exciting applications. However, unlike computer vision and speech, there is a dearth of large scale training data for developing robust and sophisticated ML models. Towards addressing this problem, this paper presents *ZeroNet* that extracts training data from publicly available videos of annotated finger gestures. Appropriate data augmentation techniques are exploited to increase the robustness and generalizability of ML models to natural patterns in user gesturing. A systematic user study with 10 users over 50 gestures demonstrates a top-1% accuracy of 82.4% and a top-3% accuracy of 94.8% with zero training overhead. While the results are promising, we believe we have only scratched the surface. Exploiting the availability of large scale video datasets that are publicly available can enhance the start of the art in a number of applications including augmented reality, virtual reality, healthcare and rehabilitation etc.

Chapter 7
mm4Arm

7.1 Introduction

Wireless signals, which are mainly used for communication networks, also have the potential to extend our senses, enabling us to see behind closed doors and track moving objects through walls [41, 253]. Accordingly, there is a growing interest in the community recently to develop novel IoT applications for sensing by exploiting radio frequency signals [254–256]. Given the compact size of modern wireless devices, this enables ubiquitous applications in the areas of smart healthcare, sports analytics, AR/VR etc. Specifically, as these signals travel in the medium, they traverse occlusions and bounce off different objects before arriving at a receiver; hence, the reflected signals carry information about the environment. By exploiting this property, this paper shows the feasibility of tracking precise 3D finger motion using mmWave signals that are popularly used in 5G networks.

Motivation and Application: This paper presents *mm4Arm*, a system that quantifies the performance of finger motion tracking for interactive applications using mmWave signals through a carefully designed simulation and measurement study. We considered using mmWave signal because FMCW-based radars are being used for ubiquitous applications in the areas of smart healthcare [257], sports analytics, AR/VR [258], autonomous driving [259], etc. Similar to the popular Google Soli platform [60], our main motivation is to enable wearable, mobile computing, and AR/VR applications where conventional touch interaction may be hard. Finger motion-based interfaces over the air are known to be a popular form of human-computer interaction [260, 261]. In contrast to Soli, which can only detect 11 predefined gestures, *mm4Arm* can perform arbitrary 3D motion tracking, thus allowing highly precise control. Decades of prior research have shown that such a finer control can enable rapid and fluid manipulation for highly intuitive interaction [262]. The finer precision of control can be

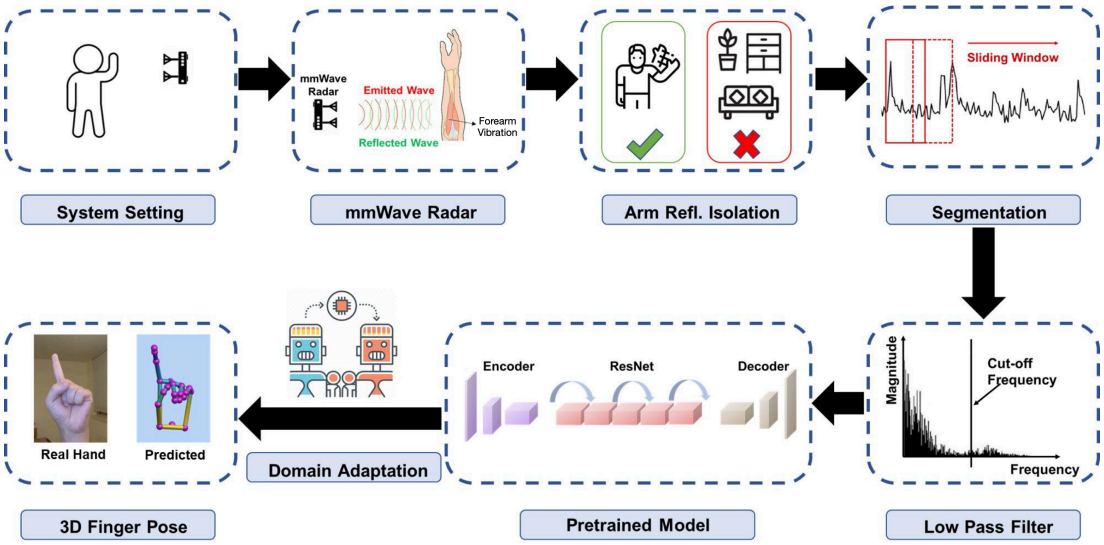


Figure 7.1: mmWave reflections are captured from the surroundings from which the *phases* of arm reflections are first isolated. After subjecting the *phase* measurements to preprocessing techniques like low pass filters, deep learning based models are designed for extracting 3D finger motion from the *phase* data. Domain adaptation is incorporated in the design for decreasing the training overhead.

observed in the case of a fluid expert interaction with hand tools (e.g. watchmaker). We believe *mm4Arm*'s accuracy can allow such a finer control. Therefore, regardless of the application, we focus on enabling the core motion tracking framework by solving the underlying challenges. We envision interesting applications of *mm4Arm*, such as developing prosthetic devices for amputees considering that forearm vibrations remain intact despite amputations [22, 81, 232], and discuss them in Section 7.9. We leave a thorough investigation of the application space for future research.

Radio Frequency (RF) Sensing vs. Vision: Recent works [31, 32, 150] track 3D finger motion using cameras placed in the environment. Powered by the latest advances in machine learning combined with the availability of large-scale training data, precise tracking is possible. However, cameras are susceptible to occlusions, lighting conditions, and interference from objects in the background. Furthermore, they are known to suffer from privacy concerns [263]. In contrast, RF sensing based on mmWave signals as performed by *mm4Arm* can be privacy-preserving and agnostic to lighting, resolution, and ambient conditions. Furthermore, RF sensing can work through materials and non-line-of-sight conditions, allowing it to be embedded into devices and environments.

Tracking Fingers by Observing the ForeArm: In this paper, we not only focus on tracking the 3D finger motion using mmWave reflections, but based on observations via simulations and measurements, we also identify the underlying conditions that enable precise tracking.

A critical observation is that the small size of fingers does not provide stable reflections to the level required for tracking. However, the data-driven analysis reveals that it is possible to indirectly track fingers by measuring reflections from the forearm. Finger motion activation involves neuro-muscular interactions, which induce minute muscular motions in the forearm. Such muscular motion produces vibrations in the forearm. Thanks to the short wavelength of mmWave signals, the *phase* measurements are extremely sensitive to small vibrations (up to $0.63\mu m$ [264]), thus opening up opportunities for precise motion tracking. Moreover, the forearm offers a rich texture and curvature and a much bigger surface for reflections, in contrast to the small size of fingers, which facilitates robust tracking. *mm4Arm* analyzes such forearm vibrations for 3D finger motion tracking.

We reiterate two critical observations made in this paper: (i) When 3D finger motion tracking is of interest in contrast to predefined gesture classification, the reflections obtained directly from fingers do not provide sufficient information. Very few reflections come back to the radar due to the small size of fingers and dominant specular reflections. A similar observation on specularity has been made earlier in the context of autonomous cars [265, 266]. (ii) Vibrations in the forearm during finger motion can capture rich information. Because of the large surface of the forearm and its curvature, the reflections are more stable and robust to natural variation in arm position, height, and orientation. This can be leveraged for 3D finger motion tracking.

Contrast with Key Prior Work: As noted earlier, prior works on finger motion tracking with radar devices are limited to discrete gesture classification. Google Soli [60] exploits reflections from mmWave signals in combination with deep convolutional and recurrent neural networks to track 11 finger motion gestures. mmASL [267] shows the feasibility of detecting 50 ASL gestures using reflections of mmWave signals. mHomeGes [268] uses mmWave signals for tracking 10 hand gestures for user interface applications in settings like smart home. RFWash [53] makes a creative use of mmWave radars for detecting hygienic methods of handwashing and alerting users accordingly. In contrast to gesture and activity classification where the search space is 10-50 predefined discrete classes, *mm4Arm*'s search space is a continuous space of 3D finger motion with 21 *degrees of freedom*. The 3D finger locations predicted by *mm4Arm* can serve as inputs to any gesture classification problem – independent of a specific application. To the best of our knowledge, *mm4Arm* is the first work to perform continuous 3D finger motion tracking using RF signals.

Challenges and Opportunities: Performing 3D finger motion tracking by sensing forearm vibrations is non-trivial with many challenges: (i) As mentioned above, the search space for the correct hand pose is high dimensional with 21 degrees of freedom. The complexity is

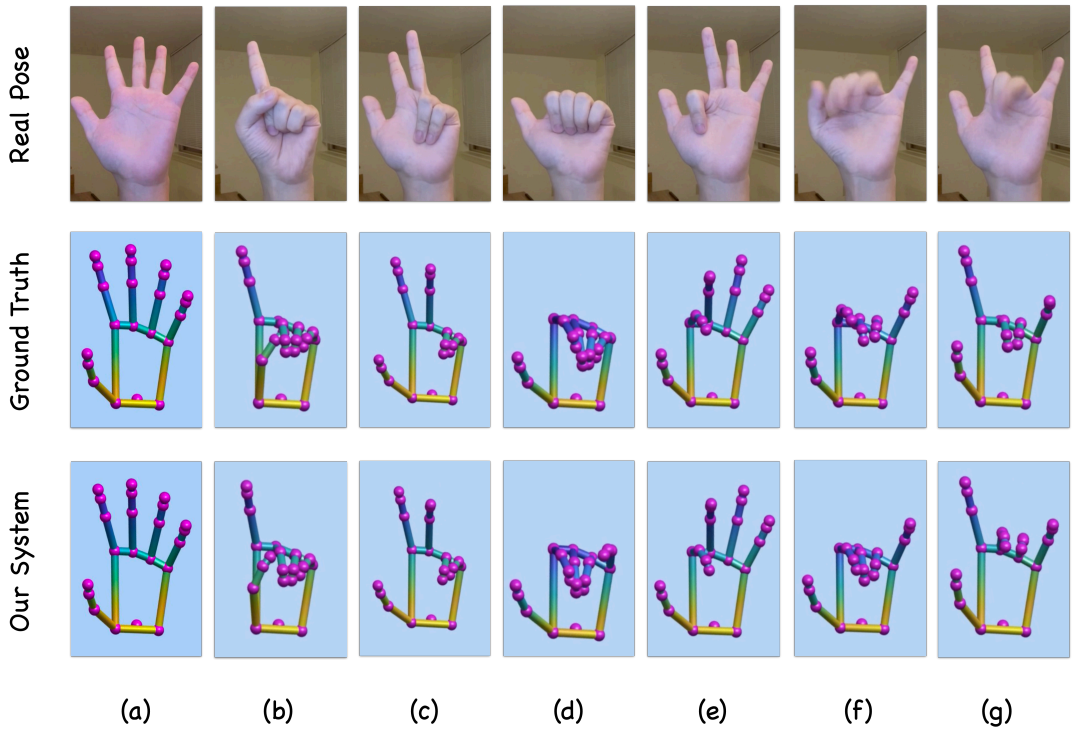


Figure 7.2: We present an approach for 3D finger motion tracking using mmWave signals. The figure shows a comparison between several real hand poses and the corresponding tracking results from a depth camera and our proposed system, *mm4Arm*. A short demo is included in this anonymous url [3].

comparable to human skeleton tracking; (ii) The vibrations due to motion of individual fingers merge into each other with complex patterns; (iii) The vibration pattern can vary among users, body sizes, anatomy, etc. While these challenges seem daunting, *mm4Arm* exploits a number of opportunities to overcome the challenges: (i) *mm4Arm* leverages anatomical constraints in finger motion towards narrowing down the search space of finger motion; (ii) Machine learning (ML) models are designed by incorporating advances in encoder-decoder and skip connections for learning the complex interrelationships between finger motions and the phase measurements while working with limits of training data availability and stability in convergence; (iii) Domain adaptation techniques are designed to develop a robust inferencing model for each user with low training overhead.

System Design: Fig. 7.1 illustrates the high-level architecture of *mm4Arm*. The radar illuminates the environment and captures reflections from the forearm and other objects in the environment. The forearm reflections are first isolated from other multipath components (wall, furniture, body, etc) based on characteristic phase variation in the forearm reflections. The phase data from forearm reflections are then preprocessed with techniques like low pass filtering for

eliminating high-frequency noise. Finally, an encoder-decoder based ML model processes the phase data and generates 3D finger motion sequences by exploiting spatio-temporal constraints of hand motion.

Implementation: *mm4Arm* is implemented using an off-the-shelf radar TI IWR6843ISK [269] operating at 60 GHz band using frequency modulated carrier wave (FMCW). The radar sensor data is pre-processed offline with MATLAB/python, and fed to ML modules implemented in TensorFlow for 3D finger motion tracking. The median error is 5.73 degrees (location error of 4.07mm), validated under a systematic user study with 10 users. The accuracy degrades gracefully with the distance of the user from the radar (evaluated upto 5ft) with robustness to environmental multipath and natural changes in arm position, height and orientation. The accuracy is also consistent under non-line-of-sight conditions and clothing. *mm4Arm* is implemented on modern smartphones - Samsung Galaxy S20, OnePlus 9 Pro – with low power consumption and a latency $\approx$ 19ms.

Contributions: We make the following contributions. (i) Feasibility of finger motion tracking by exploiting reflections from the forearm (ii) Free-from 3D finger motion tracking for arbitrary hand motion with mmWave radar. (iii) Design of ML models that fuse anatomical constraints of finger motion with sensor data for accurate 3D finger motion tracking. (iv) A systematic validation with 10 users and implementation on embedded operating systems. Fig. 7.2 depicts some examples of *mm4Arm*'s tracking quality. A short demo is included in the anonymous url [3].

7.2 Background

We begin with a brief overview of: (i) Relationship between finger motion and forearm vibration. (ii) Anatomical constraints of the human hand to be incorporated in ML models for narrowing down the search space for 3D finger motion tracking.

7.2.1 Relationship between Finger Motion and Forearm Vibration

Muscles responsible for motion of fingers are located in the forearm (Fig. 7.3). Depending upon which fingers and the manner in which they need to move, a unique pattern of muscles in the forearm are activated, thus inducing minute vibration in the forearm. *mm4Arm* tracks such forearm vibrations for 3D finger motion tracking. We now provide a brief overview of forearm muscular involvement during finger motions. Several muscles are involved in performing finger motions. Fig. 7.3a and 7.3b depict the anatomical structure of the forearm where the muscles

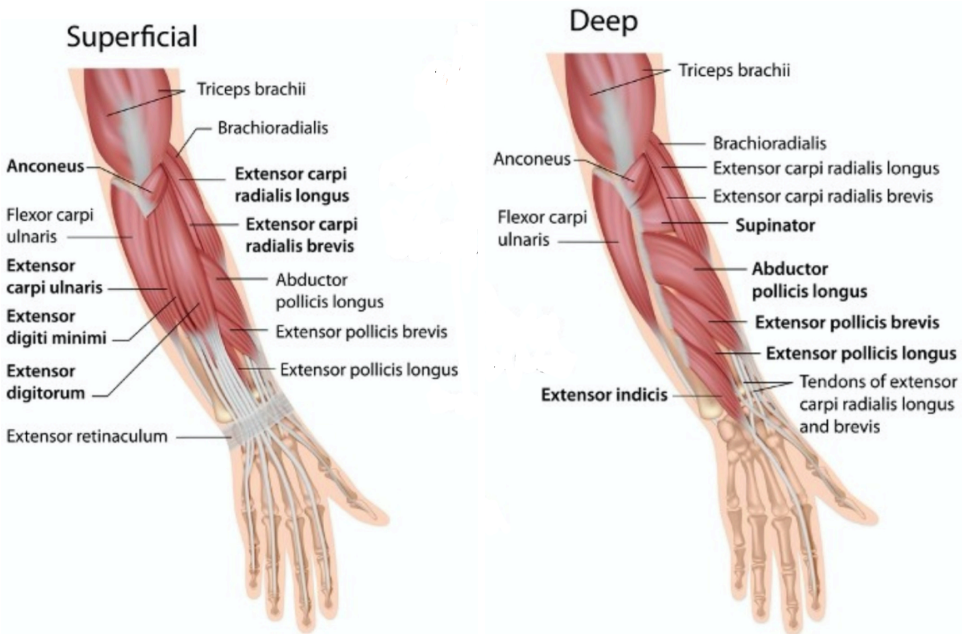


Figure 7.3: Muscles responsible for finger motion are located in the forearm [11]. Movement of these muscles causes the forearm to vibrate during finger motion.

move. *Extensor Pollicis Longus* extends the thumb joints whereas *Abductor Pollicis Longus and Brevis* performs thumb abductions. *Extensor Indicis Proprius* extends the index finger. *Extensor Digitorum* extends the four medial fingers and *Extensor Digiti Minimi* extends the little finger. *Volar interossei* and *Dorsal interossei* group of muscles are responsible respectively for adduction and abduction of index, ring, and little fingers towards/away from the middle finger. They are connected to the *proximal phalanx* and *Extensor digitorum*. Other muscles that are involved in large scale motion and supporting strength include *Supinator* for forearm motion, *Anconeus and Brachioradiali* for elbow joint, *Extensor Carpi Ulnaris*, *Extensor Carpi Radialis Longus and Brevis* for wrist joint etc. The motion of these muscles in the forearm induces vibrations in the forearm. The pattern of vibration is a function of what muscles need to move to activate a specific finger motion pattern. *mm4Arm* exploits such forearm vibrations for tracking 3D finger motion.

7.3 Overview of the Experimental Platform: mmWave Radar and FMCW

mm4Arm adopts an FMCW radar for tracking forearm vibrations. An FMCW radar works by emitting *chirps*. The *chirp* is reflected back by objects in the environment and based on the time differences between transmission and reception of *chirps* and the doppler shifts, the radar

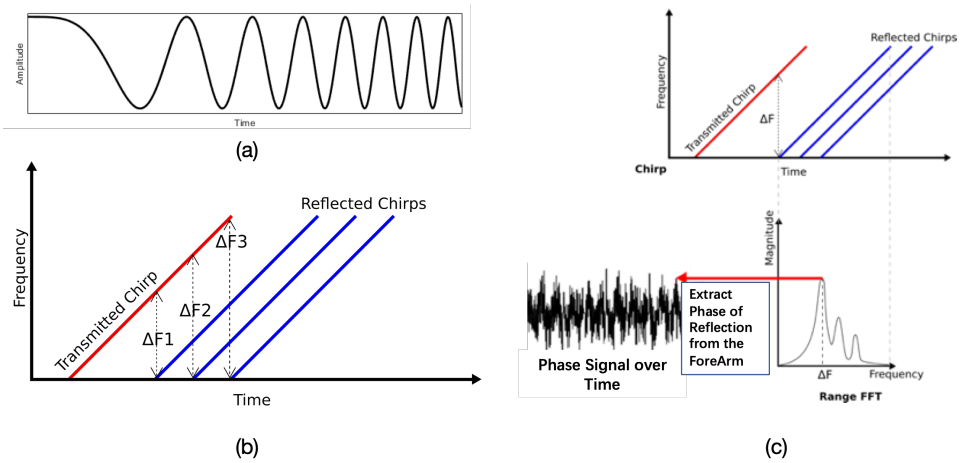


Figure 7.4: (a) An FMCW signal with linearly increasing frequency (b) The reflected FMCW signals from objects in the environment (c) A range-FFT will result in multiple *peaks* corresponding to objects in the environment. Tracking the phase of the *peak* from forearm reflections will facilitate finger motion tracking

can estimate the *range* (distance) of these objects and velocities.

A *chirp* and the working principle of FMCW radars are visualized in Fig. 7.4a, which shows a sinusoidal signal with linearly increasing frequency, which is employed by TI *IWR6843ISK* [269] radar, used in *mm4Arm*. Since the transmitted signals are frequency-modulated signals, the reflected components will also be frequency-modulated signals. However, because they are delayed, at any given point in time, there is a constant frequency difference between the transmitted and reflected *chirp* as depicted in Fig. 7.4b. By computing the frequency difference ΔF between the transmitted and received chirps, the distance of the reflecting object can be computed. The below equation precisely converts the frequency difference into the *range* (r) of the object from the radar.

$$r = \frac{\Delta F}{Slope} \tag{7.1}$$

where $Slope$ refers to the rate at which the chirp frequency is linearly modulated.

Depicted in Fig. 7.4b, multiple reflected chirps from different multipath components can be received at the radar. By performing an FFT operation (called *range FFT*), different multipath components, and their *ranges* can be determined (Fig. 7.4c). The resolution at which *ranges* can be computed can be expressed as a function of the chirp sweeping bandwidth B as follows [270]:

$$\Delta R = \frac{c}{2B} \tag{7.2}$$

where c is the speed of light. If the entire working bandwidth of the radar ($3.705GHz$) is effectively swept by a chirp, the above equation predicts a range resolution of $4.05cm$. While this is good for applications like human activity recognition (running, sitting, etc.) where the motion of objects is at larger scales, the resolution is not sufficient for tracking minute micrometer-level vibrations needed for capturing the forearm vibrations during finger motion. Towards capturing a higher resolution range information, *mm4Arm* exploits the *phases*. The *phase* variations can capture minute changes in motion of the reflector, as per the below equation.

$$\Delta\phi = \frac{2\pi\Delta r}{\lambda} \tag{7.3}$$

Given that the wavelength is in the order of millimeters ($\approx 4mm$), and a typical phase noise of $0.057°$(based on our experimental observations and comparison with the ground truth of the phase error), extremely small changes in range ($\Delta r \approx 0.63$ *um*) can be detected from *phase* variations. *mm4Arm* tracks such variations to sense minute vibrations in the forearm. Fig. 7.4c depicts extraction of continuous phase changes from the radar.

The *range-FFT* will result in multiple *peaks* due to multipath reflections. Among these *peaks*, the *peak* corresponding to reflection from the arm is first isolated (Section 7.5.1). By measuring the phase of this FFT *peak*, and tracking its variations continuously over time and across antennas helps identify rich patterns which are predictive of 3D finger motion.

7.4 Electromagnetic Simulations of Feasible Reflections

We conduct simulations using Remcom WaveFarer toolkit [271] in the 60 GHz spectrum to understand what reflections are feasible for finger motion tracking. WaveFarer uses shoot and bounce ray-tracing technique [272] in addition to techniques based on physical optics [273] (for computing scattered fields), method of equivalent currents [274] (for diffraction effects), and uniform theory of diffraction [275] (for multipath effect between objects). This enables highly accurate simulations [276]. Such techniques have been successfully used in radar sensing applications, such as autonomous driving [277]. Using this platform, we emulate the Texas Instruments *IWR6843ISK* radio [269] at 60 GHz (used by *mm4Arm*) and place it in front of a

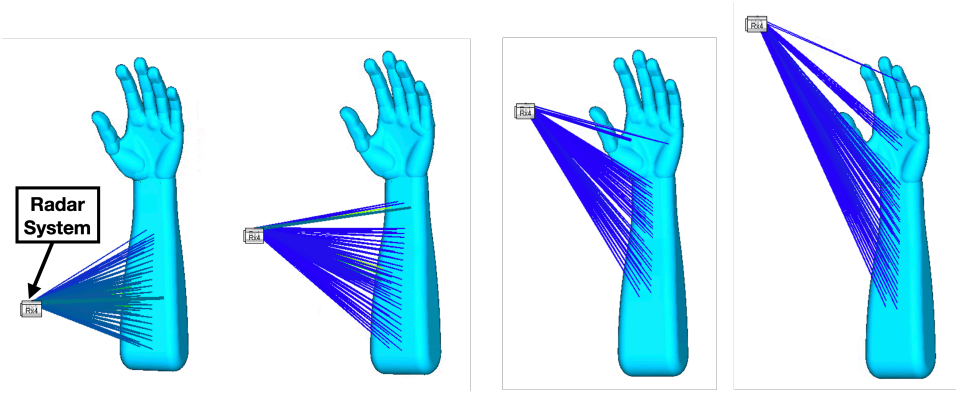


Figure 7.5: Simulation results at 60 GHz: The blue lines visualize the rays that are transmitted and returning back to the radar via reflections. We vary the height of the radar to observe all surfaces on the hand that can yield stable reflections back to the radar. Results indicate that the reflections from fingers are negligible even when the radar is placed close to fingers. However, the large surface combined with texture and curvature of the forearm provides stable reflections for 3D finger motion tracking.

CAD model of a human arm to obtain a preliminary assessment of feasibility of reflections. The results are elaborated next.

Lack of stable reflections from fingers: Fig. 7.5 visualizes the simulation results of reflected rays that arrive at the radar. Evidently, very few reflections from the palm and fingers appear at the radar, even when the radar is placed close to the fingers. We observe that this is mainly because of the small size of fingers coupled with the specular nature of the dominant reflections that deflect the rays into random directions. A similar observation on specularity is made in [265, 266], in the context of applications including autonomous cars. Also validated by real experiments in Section 7.6, the ML models to capture 3D finger motion using such reflections result in very high errors. Therefore, we seek alternative approaches for tracking the motion of fingers.

Stable reflections from the forearm: While the small size of fingers do not provide stable reflections, we observe that there is an opportunity to indirectly track fingers by focusing on forearm reflections. Finger motion activation involves neuro-muscular interactions that trigger minute muscular motions in the forearm, which in turn will induce vibrations in the forearm. Simulation results in Fig. 7.5 also show that reflections from the forearm can be tracked reliably at the radar owing to its larger surface, texture, and curvature, which can return significant reflections back to the radar. The ability to obtain significant reflections from the forearm allows *mm4Arm* to sense forearm vibrations and hence track finger motion. Note that The mmWave signal doesn't have to penetrate through the fat in the forearm, or have to go inside the human body. We are actually measuring the surface vibration of the forearm caused by muscle activating, and that vibration can cause phase variation of mmWave measurements.

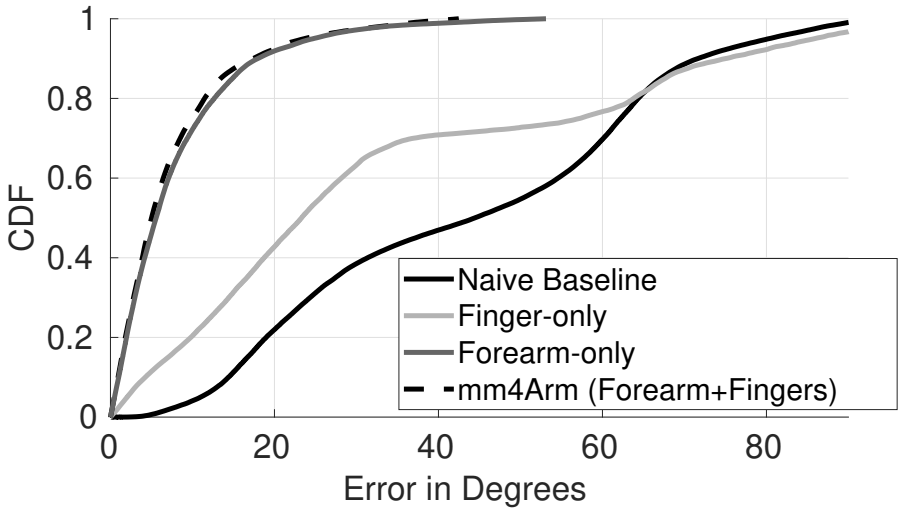


Figure 7.6: The reflection from fingers (*Fingers-only*) do not capture sufficient reflections and the accuracy is close to naively predicting an always open palm. On the other hand, forearm reflections (*Forearm-only*) can provide reliable prediction of 3D finger motion. Accordingly, ForeArm reflections mainly contribute to the high accuracy in *mm4Arm*.

Validation of Simulation Outcome via Real Measurements: In contrast to high-fidelity electromagnetic simulations, the real data does not offer fine enough resolution to visualize the individual reflections from the radar. Therefore, we only provide the end result of 3D hand pose prediction with real data (Fig. 7.6 shows joint angle errors). Towards this end, we obtain the phase measurements from the mmWave radar, which is a superimposition of phases from individual reflections, and analyze their variation over time in an attempt to capture the rich spatiotemporal relationships to predict the 3D hand pose. We employ a deep learning model for this prediction (detailed in Section 7.5.2). Specifically, we compare the following three cases: (i) *Finger-only*: Analysis of reflections from fingers-only (forearm blocked by a metal sheet) (ii) *Forearm-only*: Analysis of reflections from forearm-only (fingers blocked by a metal sheet) (iii) *mm4Arm*: Analysis of all reflections from finger and forearm. We compare these three cases with a *naive baseline* in Fig. 7.6, that always outputs the static hand pose with palm open. We make three observations from Fig. 7.6. (i) With *Finger-only*, the error is higher and closer to the *naive baseline*, indicating that finger reflections are not sufficient enough to capture 3D hand pose, (ii) *Forearm-only* results in high accuracy which is comparable to vision based approaches (evaluated in Section 7.7). (iii) This indicates that accuracy with *mm4Arm* as shown in Fig. 7.6 is mainly due to forearm reflections, and any reflections captured from fingers are too sparse to make any difference in the accuracy. The rest of the paper expands on the details of the deep learning model and associated challenges and provides a thorough performance of

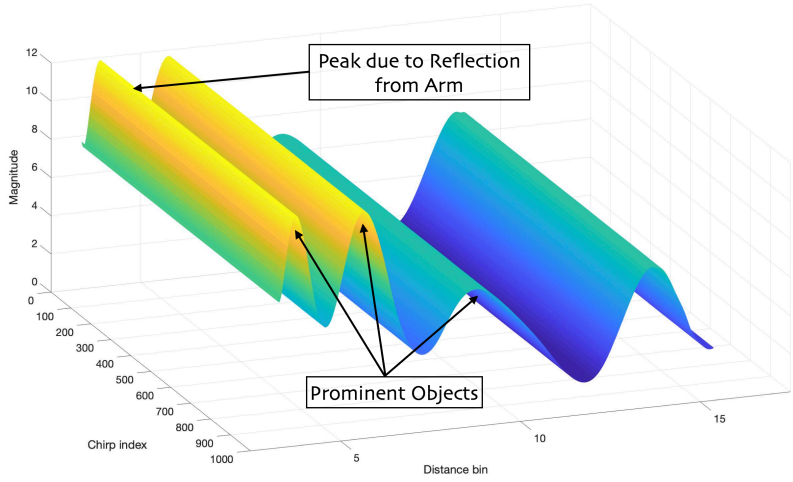


Figure 7.7: Tracking of FMCW peaks over time helps eliminate noisy peaks. The phase data corresponding to the peak from the forearm reflection is used for 3D hand pose tracking

quantitative and qualitative results via systematic implementation and measurements.

7.5 From Forearm Vibrations to 3D Finger Joint Tracking

In this section, we describe the following key signal processing and ML modules designed for tracking the 3D finger motion. (i) Isolation of arm reflection from other multipath components (ii) Machine learning model for mapping RF phase data into 3D finger motion pattern by exploiting the spatio-temporal relationships in finger movements. (iii) Domain adaptation techniques for minimizing the training overhead for new users of *mm4Arm*.

7.5.1 Isolation of Forearm Reflection

As discussed in Section 7.2, a given *range-FFT* window will include the reflection from the forearm as well as multipath reflections from other objects in the environment. We face two main challenges in isolating the arm reflections: (i) Several noisy peaks show up in the range-bin mainly because of hardware related artifacts. (ii) In addition to the noisy peaks, there will be peaks corresponding to reflectors in the environment such as walls and furniture. Towards better isolation of signal of interest from the above sources, *mm4Arm* tracks consistent peaks across successive frames. Since the noisy peaks do not consistently appear at a given distance, they

are eliminated. This step also eliminates reflections from dynamic multipath such as mobile reflectors. Fig. 7.7 shows an example where the arm reflection is consistently tracked over time. In addition to arm reflections, there also exists reflections from other objects in the environment. In the real experiment, we are able to eliminate reflections from other environmental reflections even when they are closer to the radar based on the isolation algorithm explained below. The phase of the reflection from the arm would exhibit rapid variations whereas phase from other reflectors will be somewhat monotonous.

Phase Variations of the ForeArm Reflection: The *phase* of the reflection from the arm would exhibit rapid variations whereas phase from other reflectors will be somewhat monotonous. By exploiting this property, *mm4Arm* isolates the reflections from the arm from other multipath components. Fig. 7.8 depicts an example of phase variation from the arm in comparison with phase variation from a wall reflection. The characteristic and higher level of variations in the arm can be exploited for isolating the arm reflections from other multipath components. By exploiting this property, *mm4Arm* designs a shallow convolutional neural network to first classify reflections into two classes: (i) Reflections due to forearm vibrations (ii) Reflections from other reflectors in the environment. The binary classifier provides high accuracy of 99.4%, thus isolating the forearm reflections from other reflections.

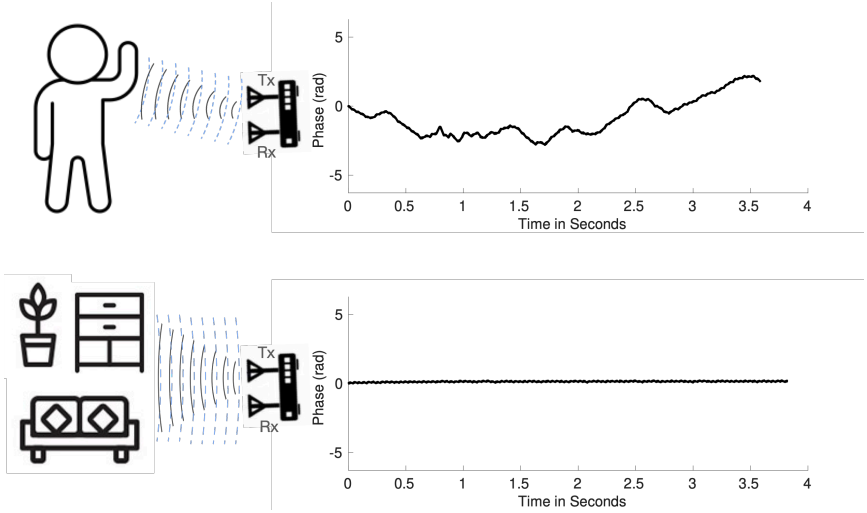


Figure 7.8: Phase variation of forearm reflections is more pronounced than the variations from other static objects

7.5.2 3D Finger Joint Tracking with Encoder Decoder Architecture

We design an encoder-decoder network as illustrated in Fig. 7.9. The size of convolution filters and the number of filters at each layer are also specified. The network is designed to capture

plausible finger pose sequences with spatial constraints across fingers with temporally smooth variations. Instead of looking at one sensor sample at a time, the network captures a holistic view of a large interval of time-series sensor data. This enables the network to enforce and learn the key spatio-temporal constraints, as well as consider historical phase data while making hand pose inferences. The network takes $2s$ of phase data as input and outputs the corresponding 3D hand pose sequence. The various components of the ML model are elaborated next.

(i) Encoder: The encoder-decoder model maps a sequence of input RF phase data to a sequence of 3D finger poses. Unlike discrete classes, the output space of the model is a continuous domain $\mathbb{R}^{21}$. Among these 21 dimensions, 5 of the dimensions (ϕ_{dip} for four fingers and ϕ_{ip} for thumb) can be directly computed using Equations since it contains the constraints between the PIP joint and the DIP joints of the 5 fingers of the human hand, which decreases 5 degrees of freedom total as each finger of hand is decreased by 1. Therefore, the actual output of the network is only 16 dimensions. The size of the input is $Y \times T$, which includes phase samples from $Y = 4$ antennas, over $T = 1000$ samples at a sampling rate of 500 Hz (2 seconds). The input first passes through an encoder network that consists of a series of convolutional layers with the input downsized at each layer with maxpool operation. The encoder attempts to capture a compact representation of the input to be used for hand pose extraction. Batch normalization is used at each layer for accelerating convergence by controlling variation in the input distribution at each layer [198].

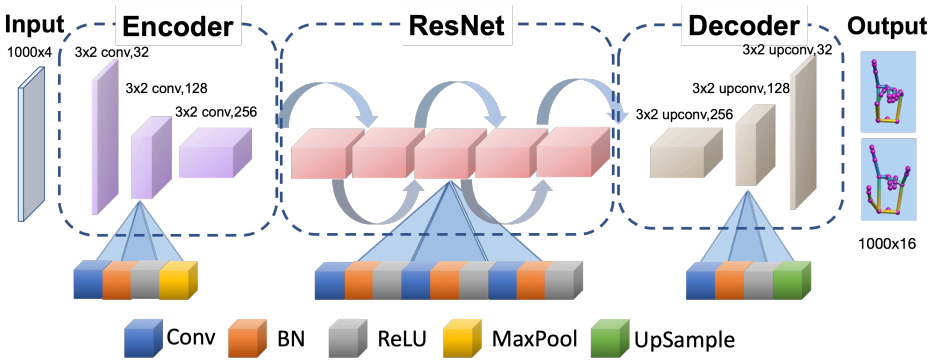


Figure 7.9: Encoder Decoder Architecture. BN = Batch Normalization

(ii) Residual Blocks: We introduce residual blocks [190] with skip connections between the encoder and decoder to increase the depth of the network. While the increase in depth allows learning stronger features, the skip connections help achieve fast convergence.

(iii) Decoder: The decoder maps the encoded representations to 3D hand pose. Upsampling layers are introduced so as to incrementally scale the size of output at each layer to eventually match the dimensions of the output. We use nearest-neighbor interpolation technique [201] for

performing upsampling. The output size is $D \times T$ where $D = 16$ is the number of joint angles predicted, and $T = 1000$ samples (2 seconds at 500 Hz).

Loss Functions and Optimization: In equations below, $\hat{\phi}$ denotes the prediction by ML models, whereas ϕ denotes training labels from depth camera (leap [30]).

$$L_{mcp,f/e} = \sum_{i=1}^{i=4} (\hat{\phi}_{i,mcp,f/e} - \phi_{i,mcp,f/e})^2 \tag{7.4}$$

$$L_{pip} = \sum_{i=1}^{i=4} (\hat{\phi}_{i,pip} - \phi_{i,pip})^2 \tag{7.5}$$

$$L_{mcp,a/a} = \sum_{i=1}^{i=4} (\hat{\phi}_{i,mcp,aa} - \phi_{i,mcp,aa})^2 \tag{7.6}$$

where $L_{mcp,f/e}$ denotes loss value of MCP joint angles with flex/extensions, L_{pip} denotes loss value of PIP joint angles, and $L_{mcp,a/a}$ is loss value of MCP joint angles with adduction/abduction. The above equations capture the mean squared error (MSE) loss in prediction of joint angles of MCP and PIP joints of the four fingers.

$$\begin{aligned} L_{thumb} = (\hat{\phi}_{th,mcp,aa} - \phi_{th,mcp,aa})^2 + (\hat{\phi}_{th,mcp,f/e} - \phi_{th,mcp,f/e})^2 \\ +(\hat{\phi}_{th,tm,aa} - \phi_{th,tm,aa})^2 + (\hat{\phi}_{th,tm,f/e} - \phi_{th,mcp,f/e})^2 \end{aligned} \tag{7.7}$$

where L_{thumb} denotes loss value of the thumb. The above equations capture the MSE loss in the MCP and TM joints of the thumb.

$$L_{smoothness} = ||(\nabla\hat{\phi}_t - \nabla\hat{\phi_{t-1}})||_2^2 \tag{7.8}$$

where $L_{smoothness}$ denotes loss value of the smoothness constraint.The above equation enforces constant velocity smoothness constraint in the predicted joint angles where ϕ_t above is a representative vector of all joint angles across all fingers at a time step t.

The overall loss function is given by the below equation.

$$L = L_{mcp,f/e} + L_{mcp,aa} + L_{pip} + L_{thumb} + L_{smoothness} \tag{7.9}$$

Note that the loss function does not include ϕ_{dip} or ϕ_{ip} because we compute them directly from

anatomical constraints.

Applying Range of Motion Constraints: The constraints across various finger joints for flex/extensions and abduction/adduction motions were described in Section 7.2. We apply such constraints to the network in order to facilitate faster learning. Towards this, we first normalize the predicted output of a joint angle by dividing it by the range constraint (for example, by 110° for ϕ_{pip}). We then apply the bounded ReLU activation (bReLU) function [202] to the last activation layer in our network. Bounded ReLU activation(bReLU) is added a upper boundary compared to normal ReLU function:

$$f_{bReLU}(x) = \min\Big(\max(0, x), 1\Big) = \begin{cases} 0 & x \leq 0 \\ x & 0 < x \leq 1 \\ 1 & x > 1 \end{cases} \tag{7.10}$$

The bReLU outputs are multiplied again with their range constraints such that the unit of the output is in degrees. The bReLU, in conjunction with other loss functions based on temporal constraints (Equation 7.8), facilitates predicting anatomically feasible as well as temporally smooth tracking results.

7.5.3 Decreasing Training Overhead via Domain Adaptation

For the encoder-decoder model proposed above, training separate models for each user will be burdensome. Therefore, we explore domain adaptation strategies to *pretrain* a model with one (*source*) user and *fine-tune* it to adapt to new users with low training overhead.

Transfer-learning based domain adaptation is popular in vision and speech processing. For example, AlexNet model [203] pretrained on ImageNet database [204] was fine-tuned for classifying images in medical domain [205], remote-sensing [206] and breast-cancer [207]. Similarly, a pre-trained BERT language model [208] was fine-tuned for tasks such as text-summarizing [209], question answering [210], etc. This significantly reduces the burden of training for a new task. In a similar spirit, we use a pretrained model from one user and fine-tune it for a different user to significantly decrease the training overhead without losing much accuracy.

The main steps in the domain adaptation process are as follows: (i) We generate a model for one user (*source*) by extensively training the model with labeled data from that user – known as the *pretrain*ed model. (ii) We collect small training data with only few labels from the new (*target*) user. Instead of developing the model for the *target* user from scratch, we initialize

the model weights to be same as the *pretrain*ed model. (iii) We make all layers in the model untrainable except certain layers which are made trainable (elaborated next). Using the few labels from the *target* user, we update the trainable layers to minimize the loss function. This is called *fine tuning*. The model thus generated will be used for making inferences on the *target* user. We explore three different approaches for the choice of trainable layers as elaborated next.

Adapting the Batch Normalization Layers: *Finetuning* the BN layers can help contain wide oscillations in the distributions of input fed from one layer to the next. Given the sufficient success in BN layers (with only a few parameters) for accelerating convergence by minimizing such *covariate shift* [198], we exploit them towards domain adaptation as well. The BN layers will learn to sufficiently transform the distribution from *target* user to a distribution of the *source* user on which the model is *pretrained*. Such a strategy has been exploited for image processing applications [278, 279]. If successful, the *pre-trained* model from the *source* user can be used for performing inferences on the target user with the *finetuning* steps discussed here.

Fine Tuning the Last Layers: Retraining the last layer of the network for a new task, while freezing the pre-trained layers from the rest of the network from another task is a popular approach with applications in image and speech processing [207, 280]. The key intuition is that a network learns meaningful representations through all layers leading upto the last few layers. Thus the initial layers are frozen during the domain adaptation. The last layers are retrained to take the representation computed by the frozen layers and compute the final output. We explore this strategy by only retraining the last layer in Fig. 7.9 for adapting a pre-trained model from one user for performing inferences on a new user.

Fine Tuning Whole Model: We continue to update the weights of the model pre-trained from another user (without freezing any weights) with limited amounts of training data from the target user. While fine-tuning the whole model might be problematic since the parameter space can be huge, because of high-level similarity in the forearm structure among humans, our experiments suggest that we do not face any issue with convergence. Prior studies also show that fine-tuning the whole network might work for some domains, as has been validated with PatchCamelyon dataset [281] for an image classification problem [282]. With fine-tuning the entire network, our model converges well with improved accuracy with limited amounts of training data. We also note that the accuracy saturates quickly with small amounts of training dataset. Detailed evaluation, and comparison with other strategies on domain adaptation discussed above, is provided in the next section.

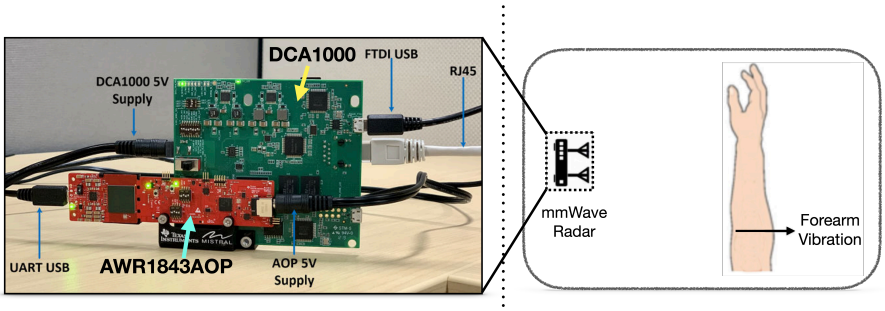


Figure 7.10: Experimental setup: Detailed view of *IWR6843ISK* radar and DCA1000EVM board for data collection (Left) [12]. Forearm vibration detection by the radar (Right)

7.6 Experimental Setup, User Study, and Implementation

We validate *mm4Arm* based on a systematic user study to analyze the performance across users, distance, multipath environments, joint angles, natural variations in forearm position/orientation, etc. This section details the data collection, size of data for training and testing across settings.

7.6.1 Data Collection and User Study

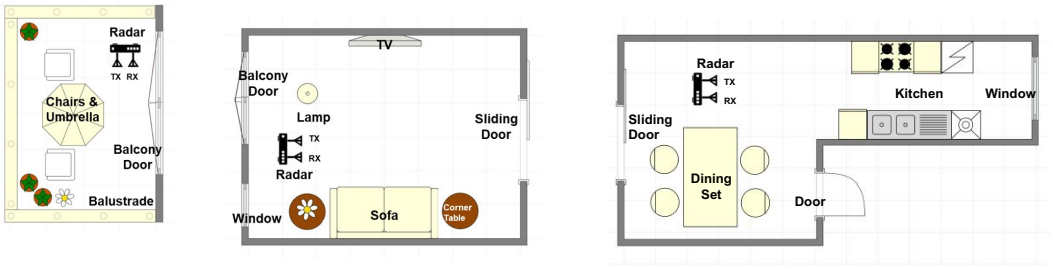
The experimental setup is depicted in Fig. 7.10. We explain the details in this subsection.

Radio Frequency Frontend: *mm4Arm*'s frontend includes Texas Instruments *IWR6843ISK* [269] mmWave radar operating in 60-64 GHz spectrum. Operating with an FMCW bandwidth of $3.705GHz$, we use the DCA1000EVM [283] platform to extract samples at 2 Msps, and obtain reflections from the human arm. The extracted phases are further low pass filtered and down-sampled to a sampling rate of 500 Hz. The phases extracted from the reflections are used for 3D hand pose tracking. Because the technology depends on forearm vibration sensing, it is important for the radar to have the visibility of the forearm. While the radar does not need to be exactly perpendicular, and it is robust to some variation and arm orientations and height, the accuracy can break down if the forearm is not clearly visible to the radar. However, even with the current setting and a number of applications such as wearable, mobile computing, and AR/VR applications where conventional touch interaction may be hard. The radar beam does not need to be focused, we use off-the-shelf TI IWR6843ISK radar with its natural config, and its field-of-view is +/- 60 degrees in Azimuth and +/- 15 degrees in Elevation [284].

Data Collection Methodology: Our user study protocol has been approved by the IRB committee at our institute. We recruit 10 users (6 males, 4 females) in the age range of 22-47,

weight range of 53-94 kgs, and height range of 5.1-6.2 ft. The users face the radar with distances upto $1-5\ ft$ from the radar device as depicted in Fig. 7.10. We also conduct experiments under non-line-of-sight conditions. For stress-testing *mm4Arm* across all possible 3D finger poses, we follow the guidelines from standard computer vision literature [285]. Accordingly, while the users were allowed to perform arbitrary random finger motions, the study staff also ensured that the users perform all *base states* of possible hand poses as defined in [285]. The majority of possible hand poses are known to be one of such *base states* or transitioning between these poses [286] based on anatomical feasibility constraints. After some practice under the guidance of research staff, the users perform arbitrary finger poses as well as pass through these *base states* in random order. This ensures good convergence of the ML models as well as generalizability. There are no discrete classes of gestures since *mm4Arm* performs tracking in a continuous $\mathbf{R^{21}}$ space.

Environment: *mm4Arm* isolates the peak from the forearm reflections (Section 7.5.1). Thus, the performance is naturally robust to environmental multipath. To better validate this, we conduct the testing under three different environments as shown in Fig. 7.11 with people moving around in the environment naturally. One-third of the data is collected under each setting with distances varying from 1-5ft. We compare the results across different environments



((a)) ((b)) ((c))

Figure 7.11: Environments for evaluation of *mm4Arm* (a) Balcony (b) Living Room (c) Kitchen

where training and testing data come from different environments.

Labels for Training and Testing: The collected data includes RF phase data from the mmWave radar and the fingers' 3D coordinates and joint angles captured by leap sensor [30]. While the radar provides RF phase data for 3D pose tracking, the leap sensor data serves as the ground truth for validation and provides labels for training *mm4Arm*'s ML models. The radar and leap data were synchronized by performing three distinct hand waving patterns at the beginning of each experiment and matching the occurrence of such patterns in the leap and radar phase data. Since *mm4Arm* performs continuous finger tracking instead of discrete gesture classification, we use MSE (instead of cross-entropy) between predicted joint angles (from radar) and ground truth (from leap) as the loss function.

Training Data Collection: As discussed before, the data collection is split uniformly across three environments in Fig. 7.11. Each user participates in 5 sessions in each environment, with one session each over distances of 1, 2, 3, 4, 5 ft. This results in 15 sessions per user. Each session lasts for 300 seconds, with enough rest between sessions. The user exits the study space after a session and returns to continue with the next session. This enables the model to develop robustness to natural changes in hand position, height, and orientation which can vary across sessions.

Test Data: The above collected data is used for developing three kinds of models as described below. For all cases, the training and testing data is taken from different multipath environments (kitchen, balcony, living room). Other specifics about test cases for each model are also described below. (i) **Model with domain adaptation (*mm4Arm*):** This is the default version of *mm4Arm*, where a model for each user where a pre-trained model from a different user is taken and fine-tuned using techniques in Section 7.5.3 such that only a small fraction (90 seconds) of user-specific training data is used for developing a model for the user. (ii) **Multi-user model:** This is a user-independent model. Here, we train a model based on training data from multiple users. The trained model is directly used for inferences on a new user without any training data from the new user. (iii) **User-dependent model:** As a baseline for comparison, we compare our system *mm4Arm* that requires only 90s of user-specific training data as noted above with a *user dependent model* that requires an excessive training overhead of 1800s of training data per user. This training data comes from 6 sessions of that user (from 1-3ft) from two environments. Testing is done on the third environment. All three combinations of train/test split (across the three environments) are considered for evaluation.

Data size: We believe the training data size is sufficient. *mm4Arm* has much fewer parameters compared to some well-known network architectures, for instance, AlexNet has 61M parameters [203] while our model has only 2M parameters. Our data size is also comparable with regular vision solutions since our input frequency is 500 Hz and we require an excessive training overhead of 1800s of pre-training data per user, while vision based dataset usually have a FPS of 30-60Hz. Moreover, we have also applied skip connections in the residual blocks of our network, helping the model to converge with fewer data samples.

7.6.2 Implementation

mm4Arm is implemented on a combination of desktop and smartphone devices. The ML model is implemented with TensorFlow [215] packages and the training is performed on a desktop with Intel i7-8700K CPU, 16GB RAM memory, and Nvidia GTX 1080 GPU. We use

the Adam optimizer [216] with a learning rate of 1e-3, β_1 of 0.9, and β_2 of 0.999. To avoid over-fitting issues that may happen in the training process, we apply the L2 regularization [217] on each CONV layer with a parameter of 0.05, and also add dropouts [200] with a parameter of 0.05 following each RELU activation. We apply anatomical constraints to the network in order to facilitate faster learning. Towards this, we first normalize the predicted output of a joint angle by dividing it by the range constraint, then apply the bounded ReLU activation (bReLU) function to the last activation layer in our network. Our system is implemented on a combination of desktop and smartphone devices. Once a model is generated from training, the inference is made entirely on smartphones using TensorFlowLite [10]. We perform the evaluation on OnePlus 9 Pro and Samsung Galaxy S20 smartphones.

7.7 Performance Results

This section provides a systematic evaluation of *mm4Arm* based on insights gathered from the simulations, the experimental platform, and the corresponding data collected as elaborated in the previous section. First, we summarize our findings and later expand on the details.

- The reflections captured directly from fingers are negligible, whereas the prominent reflections generated from the forearm can provide reliable accuracy. The median error in joint angle tracking is 5.73°. The median error in location is $4.07mm$.
- *mm4Arm* can track fingers under non-line-of-sight conditions and even when the forearm is occluded from wearing clothes. The accuracy is robust to user diversity and natural variation in arm position, height, and orientation. The multipath environment does not impact the accuracy since the forearm reflections are isolated from other multipath components.
- *mm4Arm* can track all finger joints as well as flex and abduction motions reliably.
- The model trained on left hand can be easily transferred for performing inferences on the right hand. This is a key result with applications in the development of prosthetic devices for amputees based on *mirrored bilateral training* (elaborated in Section 7.9).
- *mm4Arm* runs on smartphones with a latency of $\approx$ 19ms and low power consumption.

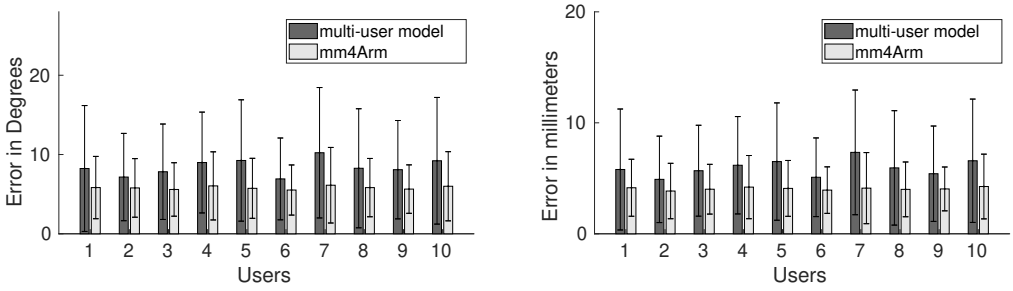
Unless explicitly stated otherwise, the reported results are obtained under the following conditions: (i) The *model with domain adaptation* as described above is used. This is the default

version of *mm4Arm*. The user-independent case is separately evaluated under *multi-user models* (Fig. 7.12). The performance of user-dependent models is shown separately (Fig. 7.18b). (ii) Results from data collected over a distance of 1-3ft from the radar are included. Other results corresponding to 4ft and 5ft are discussed separately in Fig. 7.16a. (iii) The errors reported are for flex/extension angles as they are prone to more errors with a high range of motion. Errors for abduction and adduction are discussed separately (Fig. 7.16d).

To give a brief overview of our evaluation, we have a summary of the measurement results: Fig. 7.12 depicts the accuracy as a function of different users averaged across all joint angles. Fig. 7.13a depicts a setting where a user is wearing a long sleeve jacket. Fig. 7.13b shows the non-line of sight setting where the hand is hidden behind a room divider wall. We also evaluate such robustness in Fig. 7.15a with users across 9 sessions at distances of 1-3ft over 3 environments. Fig. 7.15b provides a breakup of accuracy over different heights of the forearm measured relative to the radar. Fig 7.15c depicts the accuracy breakup over different settings. Fig. 7.16 depicts the accuracy vs distance, different fingers, finger joints and abduction/adductions and flex/extensions. Fig 7.18a,b depicts Accuracy comparison of different domain adaptation techniques and size of training data. Based on these results, the factor that may obviously affect the accuracy includes distances between radar and users' forearm and domain adaptation techniques.

Qualitative Results: A short demo is included in this anonymous url [3]. Fig. 7.2 shows samples of 3D finger motion tracking. We visualize the real hand with the corresponding tracking by the leap sensor (ground truth) and our system *mm4Arm* (mmWave radar). Tracking by *mm4Arm* closely follows the leap ground truth and the real hand. Samples in (f), and (g), indicate that the tracking is consistent even when the fingers are moving. The overall results suggest that *mm4Arm* can track the 3D finger motion pattern with good accuracy.

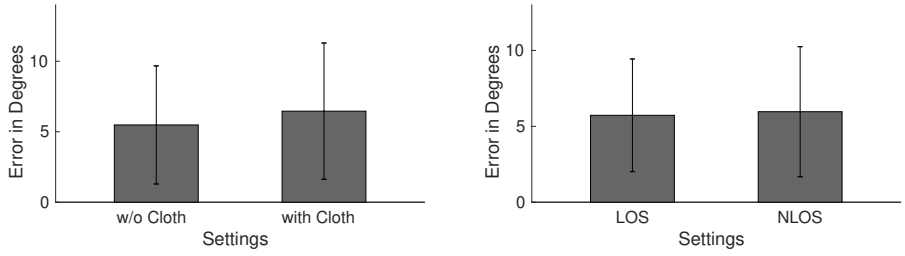
Overall Accuracy vs Users: Fig. 7.12 depicts the accuracy as a function of different users averaged across all joint angles. While the multi-user model where the training data is generated from 9 users and tested on an unknown user performs well with a median error of $8.47°$, the tail errors and the deviation can be large. Domain adaptation in *mm4Arm* can dramatically cut down the tail errors in addition to improving the median error to $5.73°$ (location error of 4.07mm), thus leading to overall better accuracy, which is comparable to vision based systems [31, 287]. Note that the location error is calculated by averaging the joint location difference of predicted joint location and the ground truth joint location. The ground truth joint locations can be fetched from Leap API [30]. The predicted joint locations can be calculated by predicted joint angles, which is the output of our model, assuming we have the information of users' finger lengths. The accuracy is consistent across users, gender, body sizes, etc.



((a)) ((b))

Figure 7.12: *mm4Arm* with domain adaptation outperforms *multi-user model* for all users: (a) Error in degrees (b) Error in millimeters

Non-Line-of-Sight Setting and Clothing: Fig. 7.13(a) (last bar) depicts a setting where a user is wearing a long sleeve jacket (Hanes Full-Zip Eco-Smart Hoodie [288]) so that the forearm is not directly visible to the mmWave radar. The accuracy does not affect much because the thickness of the clothing material is typically much smaller to cause any significant attenuation of mmWave signals. Similarly, Fig. 7.13(b) shows the non-line of sight setting where the hand is hidden behind a room divider wall (YASRKML 3 Panel Room Divider [289]), with a distance of 3ft between the radar and the hand. The chosen material is similar to typical materials used for partitioning indoor spaces. The median error under this setting is $5.97°$ whereas the median error under line-of-sight setting at the same distance was $5.73°$. This indicates the basic feasibility of sensing under non-line-of-sight conditions. Therefore, the sensing device can be embedded into materials and environments thereby enhancing the ease of deployment.



((a)) ((b))

Figure 7.13: (a) The accuracy remains stable when users wear long sleeve cloth (b) *mm4Arm* is robust to Non-Line-of-Sight conditions

Performance Analysis over an Application in Gesture Recognition: *mm4Arm* performs 3D tracking of finger motion in a generic context and independent of any application. The

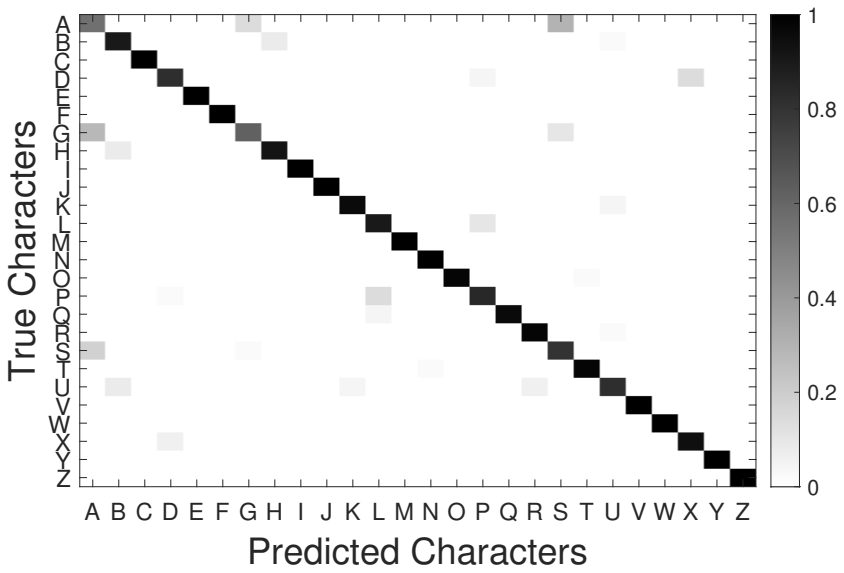


Figure 7.14: Confusion Matrix of classification.

tracking results can be used for any application. We evaluate the feasibility of *mm4Arm* over a real-world application in recognition of alphabets in American Sign Language (ASL) as defined in [290]. The classification was performed by comparing the $\mathbf{R^{21}}$ space of joint angles of the users with the joint angles corresponding to each gesture class. The gesture class with the minimum Euclidean distance from the user's finger joint angles is declared as the inferred gesture. Fig. 7.14 depicts the confusion matrix of the classification. Evidently, most gestures are classified correctly with an overall accuracy of 92.23%. Gestures such as A and S are misclassified sometimes because their hand-poses are similar. This demonstrates the feasibility of using *mm4Arm* in real-world applications.

Robustness to Variation in Arm Position, Height, and Orientation: The ML models need to be robust to natural variation in arm position, height, and orientation. We evaluate such robustness in Fig. 7.15(a) with users across 9 sessions at distances of 1-3ft over 3 environments in Fig. 7.11. The domain adaptation data for test sessions come from a different environment (and hence different session) than the training environment. As discussed in the user-study methodology, users exit the study space after each session before coming back to start a new session. This introduces natural variations in arm position, height, and orientation across sessions. Yet, the accuracy is stable across all sessions, thus indicating that *mm4Arm* is robust to the above variations. Furthermore, across these sessions, Fig. 7.15(b) provides a breakup of accuracies over different heights of the forearm measured relative to the radar. The height is

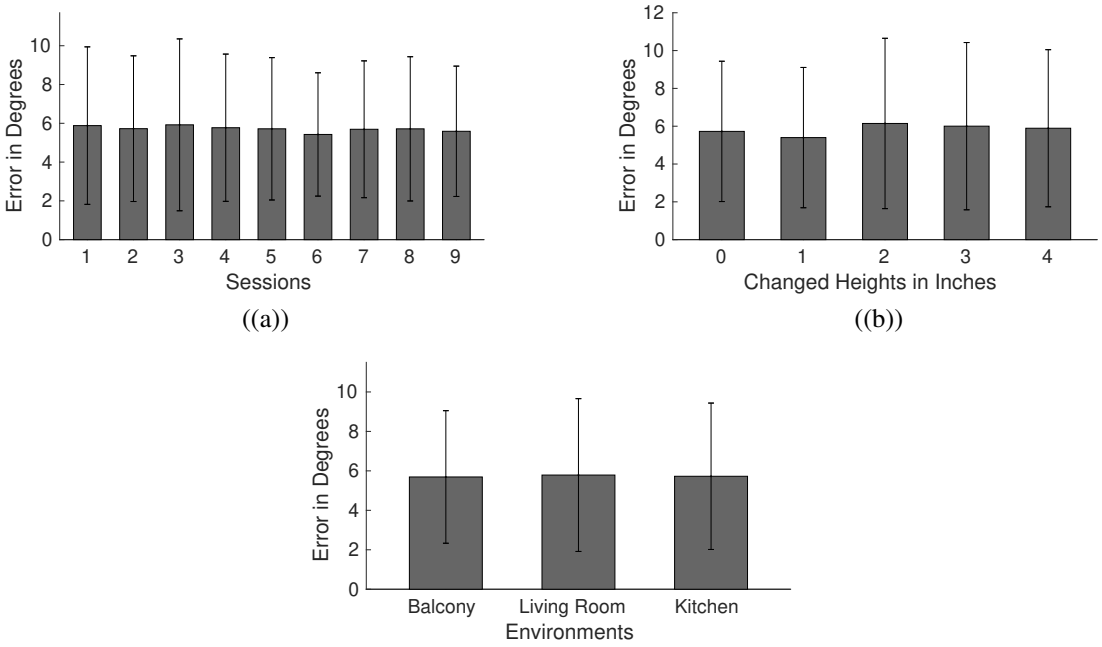


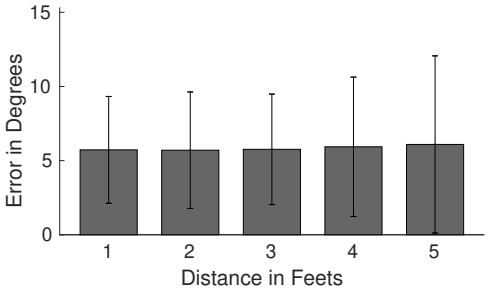
Figure 7.15: (a) Accuracy over sessions with variations in arm position/orientation (b) Accuracy over different heights of the forearm relative to radar (c) Accuracy over environmental settings.

measured from the wrist joint. When the radar is pointing directly at the wrist joint, the height is 0, and increases as we move downwards from the wrist to elbow. The accuracy is robust to the height of the arm because the training data incorporates such diversity.

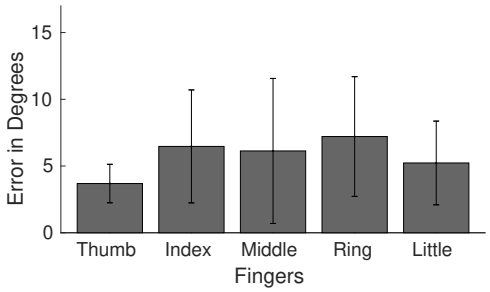
Robustness to Environmental Setting: Fig 7.15(c) depicts the accuracy breakup over different settings. Since *mm4Arm* eliminates other multipath components before further processing steps (Section 7.5.1), there would be almost no impact of multipath interference on the accuracy. The accuracy is consistent across different settings.

Accuracy vs Distance: Fig 7.16a depicts that the accuracy is almost similar for 1-3ft. However, beyond that distance, the accuracy starts to degrade gracefully. While the median errors do not show much degradation the accuracy starts decreasing in the tail. With the increasing distance of the user, the SNR of the forearm reflection decreases which results in higher tail errors.

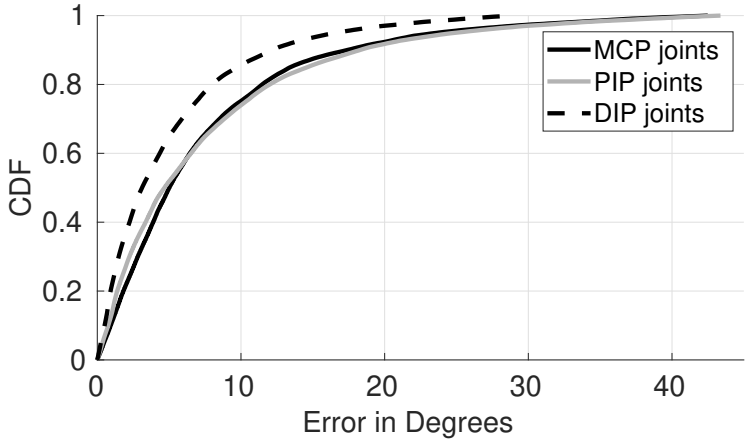
Accuracy vs Fingers: Fig. 7.16b depicts the accuracy for the four fingers and thumb. The accuracy is averaged over $\phi_{mcp,f/e}$, ϕ_{pip}, and ϕ_{dip} for the four fingers. For the thumb, the accuracy is computed over $\phi_{mcp,f/e}$, $\phi_{tm,f/e}$, and ϕ_{ip}. The ML models can accurately predict the motion of all fingers. The accuracy of thumb is slightly higher because of a smaller range of motion.



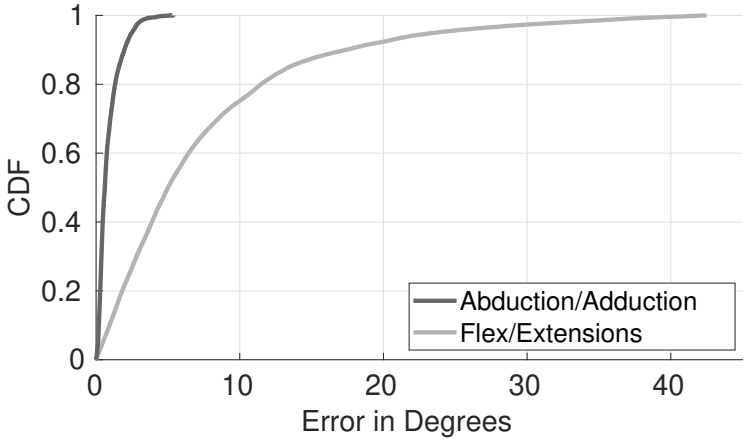
((a))



((b))



((c))



((d))

Figure 7.16: Accuracy vs (a) Distance (b) Fingers (c) Finger Joints (d) abduction/adductions and flex/extensions

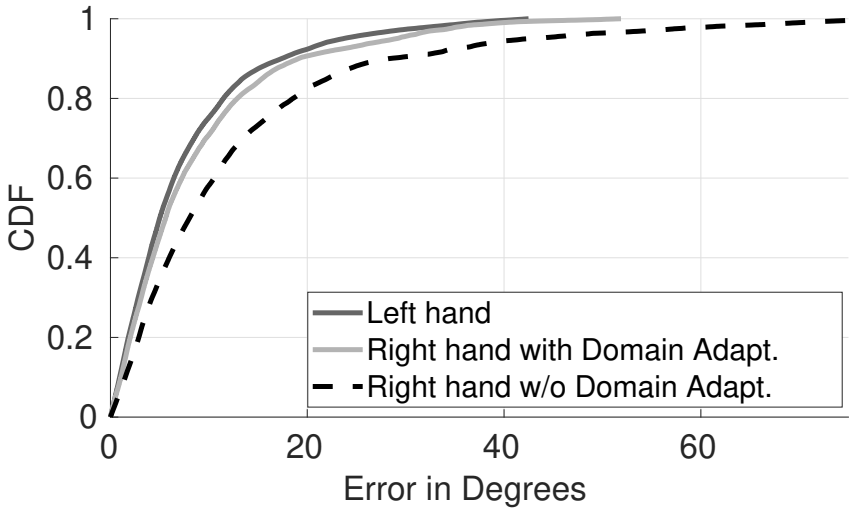


Figure 7.17: Transfer of model from left hand to the right hand.

Accuracy vs Finger Joints: Fig. 7.16c depicts the accuracy as a function of the three finger joints – $\phi_{mcp,f/e}$, ϕ_{pip}, and ϕ_{dip}. *mm4Arm* tracks all finger joints with consistent performance. Fig. 7.16d depicts the accuracy as a function of flex/extensions and abduction/adductions. Abduction/adduction angles have higher accuracy than flex/extensions because of a limited range of motion.

Transferring Model from Left Hand to Right hand: Fig. 7.17 depicts the accuracy when the model trained for the left hand is transferred for inferences on the right hand. Even without domain adaptation, the direct use of a model trained on the left hand provides good accuracy for inference on the right hand. After domain adaptation with small training data from the right hand (90s), the accuracy is comparable to the left hand. This opens up possibilities of developing ML models for amputees with missing fingers. A model can first be learnt from the hand without amputation, which can then be transferred to the hand with amputation.

Comparison of Domain Adaptation Strategies: Fig. 7.18a depicts the comparison of three domain adaptation strategies discussed in Section 7.5.3. Because of the high-level similarity in forearm-vibration pattern across users, it turns out that fine-tuning the whole model is feasible and achieves the best accuracy with stable convergence even with limited domain adaptation data. Therefore, *mm4Arm* adopts a strategy that updates the whole network during domain adaptation.

Accuracy vs Size of Training Data: Fig. 7.18b depicts the accuracy variation with the size of training data for *mm4Arm* with domain adaptation in comparison to a baseline of the *user dependent model*. With only 5% (90 s) of training data as the *user dependent model*, the

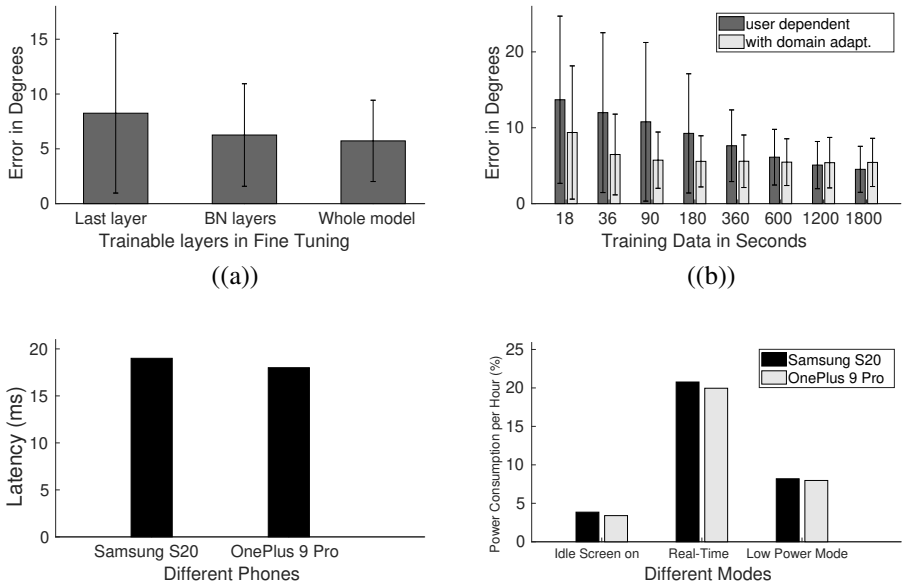


Figure 7.18: (a) Accuracy comparison of different domain adaptation techniques. (b) Accuracy vs size of training data. (c) Latency of Execution and (d) Power Consumption on Smartphones

mm4Arm with domain adaptation achieves a performance close to the *user dependent model*. Thus, *mm4Arm* can adapt to a practical setting with limited training data.

Comparison with Vision: To give an idea of how *mm4Arm* performs compared to some other solutions, we compare *mm4Arm* with SOTA camera based techniques – *Vision1* [287] and *Vision2* [31] – as shown in Fig. 7.19. We note that *mm4Arm*'s accuracy is comparable to camera-based approaches while offering other benefits over cameras such as not being privacy-invasive, agnostic to lighting conditions, and the ability to work under basic occlusions, thus allowing the *mm4Arm* system to be embedded in everyday devices and environments.

System Profiling: Latency, Power Consumption, and Processing Overhead: Fig. 7.18c depicts the latency of *mm4Arm*'s ML models on modern smartphones - Samsung S20 and OnePlus 9 Pro – with TensorFlowLite. The latency figures denote the overall time spent in processing 2 seconds of input sensor data using the encoder-decoder architecture. The latency is under 20ms on both smartphones which indicates low processing overhead. We use Batterystats and Battery Historian [220] tools for profiling the energy of the TensorflowLite model. We compare the difference in power between the following two states. (i) The device is idle with screen on. (ii) The device is making inferences using TensorflowLite model. Depicted in Fig. 7.18d, the idle display-screen on discharge rate is $3.85\%, 3.40\%$ per hour for two phones. The discharge rates while executing the ML models is also summarized in the figure. Since

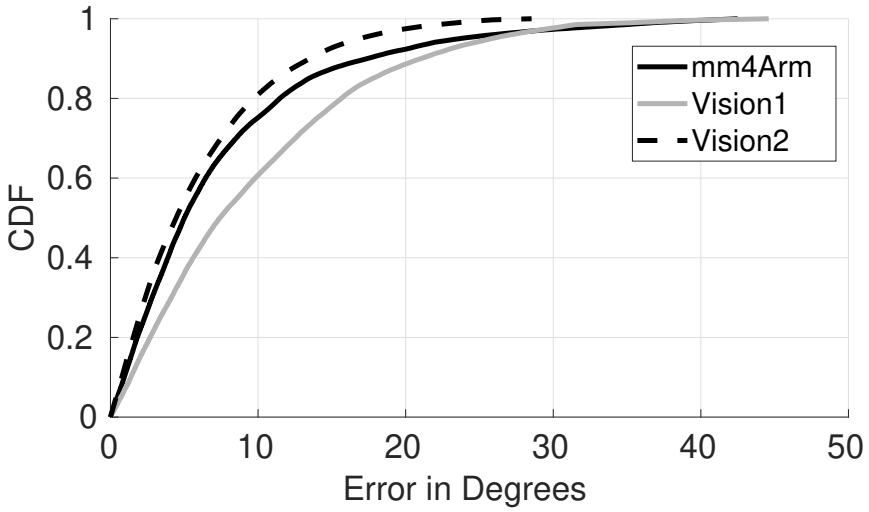


Figure 7.19: Comparison with Vision

the encoder-decoder processes data in chunks of 2s, it will incur a delay of atleast 2s if we process the data only once in 2s. Processing the model once in 2s will result in a discharge rate of $8.19\%, 7.97\%$ per hour for the two phones. Towards making it real-time, we make a modification where at any given instant of time, previous 2s segment of data is input to the network to obtain instantaneous real-time results. This provides real-time tracking at the expense of power. Depicted in Fig. 7.18d, this entails continuous/redundant processing thus increasing the discharge rate to $\approx 20.77\%, 19.96\%$ per hour for the two phones. The low-power mode trades off real-time performance (2s delay) for power savings. Depending on requirements of real-time latency or energy efficiency, a user can choose between the two modes.

7.8 Related Work

Table 7.1 provides a brief overview of *mm4Arm* in the context of key prior work. In the table, *21 DoF Tracking* refers to the ability to track all finger joints which have a total of 21 degrees of freedom (DoF). The related work falls under three categories as elaborated below. We compare and contrast the need for *mm4Arm* with respect to each of these areas.

Vision: Depth cameras like kinect [29] and leap [30] sensors can track fingers. They have revolutionized the gaming industry by gesture interfaces. Recently, even monocular RGB cameras are able to capture 3D motion of fingers by exploiting advances in ML together

Table 7.1: Scope of *mm4Arm* in the context of key prior works. To our best knowledge, *mm4Arm* is the first work that performs 3D hand pose tracking with 21 degrees of freedom using RF signals with benefits as highlighted in the table.

System Tracking	**Sensing Band**	**Robustness to Lighting**	**Non Line of Sight**	**21 DoF**
Google Soli [60]	mmWave (60 GHz)	✓	✓	✗
mmASL [267]	mmWave (60 GHz)	✓	✓	✗
RFWash [53]	mmWave (60 GHz)	✓	✓	✗
SignFi [54]	WiFi (5 GHz)	✓	✓	✗
WiSee [51]	WiFi (5 GHz)	✓	✓	✗
FingerIO [143]	Ultrasound (18-20 kHz)	✓	✓	✗
LLAP [291]	Ultrasound (48 kHz)	✓	✓	✗
GANerated [292]	Visible Light	✗	✗	✓
MediaPipe [293]	Visible Light	✗	✗	✓
Leap [30]	Visible Light and Infrared	✗	✗	✓
***mm4Arm* (This paper)**	**mmWave (60 GHz)**	✓	✓	✓

with the availability of large-scale training data [31, 32, 150]. While such works are truly transformative in nature, vision-based approaches can be privacy-invasive and susceptible to changes in lighting, background, and resolution. Digits [184] uses wrist-mounted infrared cameras for 3D finger pose tracking. Similarly, DorsalNet [294] uses wrist-mounted visual cameras for 3D finger motion tracking. FingerTrak [33] has innovatively designed wearable thermal cameras to track 3D finger motion but has issues with background temperature stability and the shifting of the camera on the hand as noted by the authors. In contrast to approaches based on external cameras, or wearable cameras, we believe *mm4Arm*'s approach provides a solution that is completely passive with robustness to lighting, resolution, and background conditions. Furthermore, *mm4Arm* can track through materials and non-line-of-sight conditions, allowing the system to be embedded into devices and environments

Radio Frequency Reflections: RF signals have been used for human body motion sensing [41–43, 295–297]. They are also used to track the motion of the hand and classify discrete gestures by using a combination of wireless channel state information (CSI), and Doppler shifts [45–47]. mmWrite [52] performs handwriting recognition using mmWave radars. RFWash [53] detects hand wash hygiene using mmWave radars near bathroom mirrors. SignFi [54] uses CSI from WiFi APs for sign language recognition. ExASL [55] tracks point clouds computed from range-doppler spectrum and angle of arrival spectrum of mmWave radars. This is used to classify upto 23 discrete gestures used in ASL. Google Soli [60] and works in [298–300] uses reflections from mmWave signals to track up to 11 finger motion gestures. *mm4Arm* differs from above works in two ways: (i) In contrast to gesture and activity classification where the search space is limited to 10 to 50 predefined discrete classes, *mm4Arm*'s search space is a continuous space of 3D finger motion with 21 *degrees of freedom* (DoF). The 3D

finger locations predicted by *mm4Arm* can serve as inputs to any gesture classification problem – independent of a specific application. (ii) *mm4Arm* senses vibrations in the forearm for 3D finger motion tracking, which results in robust tracking in comparison to reflection from fingertips that are not stable (details in Section 7.4).

Wearable Sensing: Sensor embedded gloves that use a combination of sensors like IMU, flex, capacitive, pressure, etc are popular [15, 35–37, 39]. However, wearing gloves precludes the user from performing natural and dexterous activities with fine precision as studied in recent works [40]. Localization and human body tracking projects exploit IMU, and WiFi sensors [41, 56, 221, 222, 301]. Similarly, IMU, WiFi, and acoustic signals have also been used for hand gesture recognition [64, 136, 140, 141]. FingerIO [143], FingerPing [112] use acoustic signals for finger gesture detection. uWave [142] uses accelerometers for user authentication and interaction with a mobile device. Tomo [302] uses electrical impedance tomography with 8 electrodes on the arm for performing classification of 8 gestures. Interferri [303] uses acoustic transducers for classification of 11 hand gestures Capacitative sensing has been systematically investigated by Capband [108] for recognition of 15 hand gestures. ElectroRing [304] attaches electrodes on the index finger and IMU sensors for detecting six different pinch-like finger gestures. DeepASL [305] uses wearable camera for ASL translation of sentences with 56 commonly occurring ASL words. ThumbTrak [306] detects 12 finger gestures by measuring relative distance between thumb and other fingers using proximity sensors. ZeroNet [27] extracts training data from videos to classify 50 hand gestures. In contrast to gesture and activity classification, as noted earlier, *mm4Arm* performs continuous 3D finger motion tracking. AuraRing [168], tracks the index finger precisely using a magnetic wristband and ring on index finger. In contrast to AuraRing, *mm4Arm* tracks all fingers. With a combination of deep learning techniques based on CNN, RNN, etc, prior works on EMG sensing perform classification of discrete hand poses [71–75] or track a predefined sequence of hand poses [74, 75]. The Myo armband has been used for 3D finger motion tracking [25, 307]. However, EMG sensors need calibration and warming of the skin to be in proper contact with the electrodes which can even take up to 5 minutes during each instance of wearing, leading to usability issues [193, 194, 308]. In contrast to using wearable devices, *mm4Arm*'s RF-based sensing is passive, since the user does not need to wear any sensor on the body.

7.9 Applications, Limitations, and Future Work

■ **Prosthetic Devices:** A key benefit of *mm4Arm* lies in the ability to sense finger motion directly from forearm vibrations instead of sensing from the fingers. Prior research has shown that subjects with amputation in the hand will still retain forearm muscular activity [22, 81, 82, 232], which manifests into forearm vibrations. Therefore, we plan to exploit the findings in this paper for the development of prosthetic devices for amputees by detecting forearm vibrations. However, because of missing fingers, it is non-trivial to generate training data that map phase patterns into corresponding 3D joint angles of various fingers. Towards handling this challenge, we plan to explore a *mirrored bilateral training* [22] scheme. At a high level, the forearm muscular activity (and the corresponding vibrations) are known to be similar in both hands for performing similar finger motion activities [228]. Therefore, an ML model trained with the non-amputee hand (without missing fingers) while inducing bilateral activation can potentially be used for performing inferences on the hand with missing fingers (amputated hand). The results in Fig. 7.17 shows the basic feasibility of such an approach, since the model trained on one hand can be used for performing inferences on the other hand. However, we leave a thorough investigation for future work.

■ **Touchless Interaction for IoT applications:** We believe *mm4Arm* can enable a number of touchless user interfaces such as typing on a virtual keyboard or gesture-based user interfaces. This is particularly useful for interaction with devices with small form factors such as a smartwatch, miniature IoT devices, game controls, robotic home assistants, mobile spectroscopy, etc. Security-based applications can be enabled where a user can lock and unlock an IoT device with a signature based on 3D finger motion pattern.

■ **Smart Assistants for Deaf People:** We envision a future application in accessibility. Voice assistants like Amazon Alexa and Google Home are popular, but inaccessible to the deaf community. The population of the deaf community is upward of 10 million in the US and about 466 million globally [115, 116, 309]. In this context, we believe touchless and fluid interaction enabled by *mm4Arm* with 21 DoF 3D finger motion can enable deaf people to interact with voice assistants by issuing complex commands like a natural language without being limited to a set of predefined gestures.

■ **Robotic Teleoperation:** Complex and unstructured robotic operation, especially in an unregulated environment may require human intelligence in addition to mechanical sturdiness and robustness of a robot. This might include applications ranging from controlling a home assistant robotic avatars or a robotic avatar in a dangerous industrial setting [310] in tasks including grasping and manipulating objects in complex ways. Towards this end, we believe

21 DoF finger motion tracking in *mm4Arm* can provide a solution for robotic avatar control, which is particularly useful if the control is desired from anywhere, anytime.

■ **Tracking Multiple Users Simultaneously:** While this paper focuses on tracking a single user, in principle, the algorithms presented could track concurrent changes from multiple forearms as long as they occur at different distances from the radar. The reflections from different users will fall in different range bins after the range-FFT (Section 7.3, Fig. 7.4). These reflections can be isolated from other multipath reflections (Section 7.5.1), and the phase variations of all users can be analyzed for tracking finger motion. We will conduct more studies in the future to validate this approach.

■ **Alternative ways of mapping forearm vibrations to finger motion:** We considered designing heuristics to map forearm vibrations to finger motion. To the best of our knowledge, there is no closed form mapping between muscles, forearm vibrations and finger motions, and we believe the neural network could learn the complex mapping above, for example, WR-Hand [307] tracks 3D hand poses by inference EMG data to neural network. We plan to explore alternative methods of doing this in the future.

■ **Thick obstacle object in NLoS experiments:** We did our non-line-of-sight(NLoS) experiments using a dividing wall that is commonly used in office settings (YASRKML 3 Panel Room Divider [289]), which is thinner than typical walls used to separate rooms. We plan to explore the experiment in separate rooms with a thick wall in between in the future.

■ **Potential weighting in loss functions:** Regarding the Loss function(Equation 7.9), We are inspired by prior computer vision works [31], who have a similar loss equation as ours. However, different weightings in the loss function might potentially optimize *mm4Arm*'s accuracy, and we will explore it in the future.

■ **Size of Sensing Device:** The current experimental setup is bulky. However, we note that the actual mmWave chip is only 2cm×2cm in size, and the dimensions of the antenna is 2.5cm×3cm. This can be integrated into a compact PCB with a SoC microcontroller to stream the range-FFT results from the radar to a smartphone. The development board used in *mm4Arm* is only for the 'prototypying phase' as this is the standard procedure in many IoT applications to extensively test the prototype before rolling out on a compact PCB [311]. Our future work will include testing the feasibility of such a fabrication to create a smaller sensing device.

7.10 Conclusion

Because of the ability to sense the sense environment around us, mmWave signals are being increasingly considered for sensing applications in addition to high-speed networking. Through a combination of high-fidelity electromagnetic simulations and real-world measurements, this paper shows the feasibility of sensing forearm vibrations for 3D finger motion tracking using mmWave signals. Anatomical constraints of finger motions were fused with ML advances in encoder-decoder, Resnets, and domain adaptation in achieving reliable accuracy with low training overhead. The inference is done with low processing and energy overhead on smartphones. Despite progress, we believe we have only scratched the surface. Opportunities exist for developing IoT applications on top of *mm4Arm* in the areas of touchless user interfaces, accessibility, prosthetic devices, etc.

Chapter 8
Conclusion

This thesis unfolded my intensive exploration into the fascinating domain of finger tracking, the applications of which span across diverse sectors such as sports analytics, healthcare and rehabilitation, sign language translation, AR/VR, haptics, and prosthetic control.

I began with an investigation into discrete finger gestures, particularly their application in American Sign Language translation. The development of the FinGTrAC system signified the feasibility of tracking finger gestures using a minimally intrusive wearable sensor modality. It showcased a significant stride from recognizing a limited number of gestures to hundreds, with a robust recognition accuracy of 94.2%. This progress marked the effectiveness of data preprocessing, filtering, pattern matching, and context fusion integrated into a Bayesian filtering framework.

The transition into the 3D continuous hand pose tracking space resulted in the conception of NeuroPose. Harnessing the power of wearable ElectroMyoGraphy (EMG) sensors, this wearable sensor modality was able to extract intricate details of finger motion. A systematic evaluation with a median error of just 6.24° substantiated the system's robustness. Moreover, the application of a mirrored bilateral training scheme, an extension of NeuroPose, innovatively addressed the challenges in generating training data for prosthetic devices, emphasizing the versatility of this approach.

In the realm of large-scale training data for machine learning models, ZeroNet emerged. This system leveraged the power of publicly available video data to perform inferences on Inertial Measurement Unit (IMU) sensors, another wearable modality. With a top-1 accuracy of 82.4% for recognition of 50 finger gestures, it confirmed the potency of data transformation and augmentation techniques in ensuring the robustness and generalizability of machine learning models.

Lastly, to alleviate the intrusive aspect of on-body sensors, the use of wireless sensor modality, specifically mmWave sensors, was explored. The resulting system, mm4Arm,

exploited the properties of mmWave signals for tracking 3D finger motion with remarkable accuracy and energy efficiency. This venture validated the counterintuitive yet effective strategy of focusing on the forearm instead of the fingers for stable reflections, thereby creating a system that operates under typical occlusions and non-line-of-sight conditions.

In conclusion, the research progress in this thesis has unveiled a myriad of opportunities and challenges associated with finger tracking technologies. Through the use of multiple modalities, specifically wearable and wireless sensor modalities, I developed innovative systems that improved the feasibility, accuracy, and efficiency of finger tracking. The insights gained from this multi-modal approach will be pivotal in shaping the future of finger tracking technologies and their applications, broadening their potential impact in our increasingly interconnected world. It is my hope that this thesis will serve as a foundational platform for future research, further propelling the advancement of Internet of Things applications for healthier, safer, and more convenient lives.

Bibliography

[1] LIU, Y., F. JIANG, and M. GOWDA (2020) "Application Informed Motion Signal Processing for Finger Motion Tracking using Wearable Sensors," in *IEEE ICASSP*.

[2] "Drawing hands in sign language poses," http://albaneze.weebly.com/step-3---drawing-hands-in-sign-language-poses.html.

[3] "A short video demo of our system," https://www.dropbox.com/s/sssl4e219w2c9al/3D_handpose_demo.avi?dl=0.

[4] CHEN CHEN, F. ET AL. (2013) "Constraint study for a hand exoskeleton: human hand kinematics and dynamics," *Journal of Robotics*, **2013**.

[5] LIN, J., Y. WU, and T. S. HUANG (2000) "Modeling the constraints of human hand motion," in *Proceedings workshop on human motion*, IEEE, pp. 121–126.

[6] PEÑA PITARCH, E. (2008) *Virtual human hand: Grasping strategy and simulation*, Universitat Politècnica de Catalunya.

[7] "Forearm Muscles," https://teachmeanatomy.info/upper-limb/muscles/posterior-forearm/.

[8] "Muscles Alive: Information and Data Sciences," https://www.bu.edu/ids/research-projects/muscles-alive/.

[9] "Dynamic Time Warping (DTW) algorithm for time series analysis," https://medium.com/datadriveninvestor/dynamic-time-warping-dtw-d51d1a1e4afc.

[10] GOOGLE (2019), "Deploy machine learning models on mobile and IoT devices," "https://www.tensorflow.org/lite".

[11] "Forearm Muscles : Attachment, Nerve Supply Action," https://anatomyinfo.com/forearm-muscles/.

[12] "tiuserguide," https://www.ti.com/lit/ug/spruix8/spruix8.pdf?ts=1631234356918.

[13] "Knuckleball Grip, Part 3: Depth of the Baseball," "https://knuckleballnation.com/how-to/knuckleballgrip3/".

[14] SUSANTO, E. A. ET AL. (2015) "Efficacy of robot-assisted fingers training in chronic stroke survivors: a pilot randomized-controlled trial," *Journal of neuroengineering and rehabilitation*.

[15] AHMED, M. A. ET AL. (2018) "A review on systems-based sensory gloves for sign language recognition state of the art between 2007 and 2017," *Sensors*, **18**(7), p. 2208.

[16] SCHEGGI, S. ET AL. (2015) "Touch the virtual reality: using the leap motion controller for hand tracking and wearable tactile devices for immersive haptic rendering," in *ACM SIGGRAPH Posters*.

[17] CORDELLA, F. ET AL. (2012) "Patient performance evaluation using Kinect and Monte Carlo-based finger tracking," in *2012 4th IEEE RAS & EMBS International Conference on Biomedical Robotics and Biomechatronics (BioRob)*, IEEE, pp. 1967–1972.

[18] CIHAN CAMGOZ, N., S. HADFIELD, O. KOLLER, H. NEY, and R. BOWDEN (2018) "Neural sign language translation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 7784–7793.

[19] CHEOK, M. J., Z. OMAR, and M. H. JAWARD (2019) "A review of hand gesture and sign language recognition techniques," *International Journal of Machine Learning and Cybernetics*, **10**(1), pp. 131–153.

[20] MURGUIALDAY, A. R. ET AL. (2007) "Brain-computer interface for a prosthetic hand using local machine control and haptic feedback," in *2007 IEEE 10th International Conference on Rehabilitation Robotics*, IEEE.

[21] ACHARYA, S. ET AL. (2007) "Towards a brain-computer interface for dexterous control of a multi-fingered prosthetic hand," in *2007 3rd International IEEE/EMBS Conference on Neural Engineering*.

[22] NIELSEN, J. L. ET AL. (2010) "Simultaneous and proportional force estimation for multifunction myoelectric prostheses using mirrored bilateral training," *IEEE Transactions on Biomedical Engineering*.

[23] LIU, Y. ET AL. (2020) "Application informed motion signal processing for finger motion tracking using wearable sensors," in *ICASSP 2020-2020 IEEE International conference on acoustics, speech and signal processing (ICASSP)*, IEEE, pp. 8334–8338.

[24] ——— (2020) "Finger gesture tracking for interactive applications: A pilot study with sign languages," *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, **4**(3), pp. 1–21.

[25] LIU, Y., S. ZHANG, and M. GOWDA (2021) "NeuroPose: 3D Hand Pose Tracking using EMG Wearables," in *Proceedings of the Web Conference 2021*, pp. 1471–1482.

[26] LIU, Y. ET AL. (2022) "A Practical System for 3-D Hand Pose Tracking Using EMG Wearables With Applications to Prosthetics and User Interfaces," *IEEE Internet of Things Journal*, **10**(4), pp. 3407–3427.

[27] LIU, Y., S. ZHANG, and M. GOWDA (2021) "When Video meets Inertial Sensors: Zero-shot Domain Adaptation for Finger Motion Analytics with Inertial Sensors," in *Proceedings of the International Conference on Internet-of-Things Design and Implementation*, pp. 182–194.

[28] LIU, Y., S. ZHANG, M. GOWDA, and S. NELAKUDITI (2022) "Leveraging the properties of mmwave signals for 3d finger motion tracking for interactive iot applications," *Proceedings of the ACM on Measurement and Analysis of Computing Systems*, **6**(3), pp. 1–28.

[29] "Microsoft Kinect2.0," `https://developer.microsoft.com/en-us/windows/kinect`.

[30] "Leap Motion Developer," `https://developer.leapmotion.com/`.

[31] MUELLER, F. ET AL. (2018) "Ganerated hands for real-time 3d hand tracking from monocular rgb," in *IEEE CVPR*.

[32] CAI, Y., L. GE, J. CAI, and J. YUAN (2018) "Weakly-supervised 3d hand pose estimation from monocular rgb images," in *ECCV*.

[33] HU, F. ET AL. (2020) "FingerTrak: Continuous 3D Hand Pose Tracking by Deep Learning Hand Silhouettes Captured by Miniature Thermal Cameras on Wrist," *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*.

[34] GLAUSER, O. ET AL. (2019) "Interactive hand pose estimation using a stretch-sensing soft glove," *ACM Transactions on Graphics (TOG)*.

[35] CONNOLLY, J. ET AL. (2017) "IMU sensor-based electronic goniometric glove for clinical finger movement analysis," *IEEE Sensors Journal*.

[36] LIN, B.-S. ET AL. (2018) "Design of an inertial-sensor-based data glove for hand function evaluation," *Sensors*.

[37] "CyberGlove Systems LLC," `http://www.cyberglovesystems.com/`.

[38] "Industry leading VR techology - Manus VR," `https://manus-vr.com/`.

[39] "5DT Data Glove Ultra - 5DT," `https://5dt.com/5dt-data-glove-ultra/`.

[40] RODA-SALES, A. ET AL. (2020) "Effect on manual skills of wearing instrumented gloves during manipulation," *Journal of biomechanics*.

[41] ADIB, F. ET AL. (2014) "3D tracking via body radio reflections," in *USENIX NSDI*.

[42] XUE, H., Y. JU, C. MIAO, Y. WANG, S. WANG, A. ZHANG, and L. SU (2021) "mmMesh: Towards 3D real-time dynamic human mesh construction using millimeter-wave," in *Proceedings of the 19th Annual International Conference on Mobile Systems, Applications, and Services*, pp. 269–282.

[43] SHI, C., L. LU, J. LIU, Y. WANG, Y. CHEN, and J. YU (2021) "mPose: Environment- and subject-agnostic 3D skeleton posture reconstruction leveraging a single mmWave device," *Smart Health*, p. 100228.

[44] ZHAN, C., M. GHADERIBANEH, P. SAHU, and H. GUPTA (2021) "Deepmtl: Deep learning based multiple transmitter localization," in *2021 IEEE 22nd International Symposium on a World of Wireless, Mobile and Multimedia Networks (WoWMoM)*, IEEE, pp. 41–50.

[45] LI, H., W. YANG, J. WANG, Y. XU, and L. HUANG (2016) "WiFinger: talk to your smart devices with finger-grained gesture," in *Proceedings of the 2016 ACM International Joint Conference on Pervasive and Ubiquitous Computing*, ACM, pp. 250–261.

[46] MELGAREJO, P., X. ZHANG, P. RAMANATHAN, and D. CHU (2014) "Leveraging directional antenna capabilities for fine-grained gesture recognition," in *Proceedings of the 2014 ACM International Joint Conference on Pervasive and Ubiquitous Computing*, ACM, pp. 541–551.

[47] SHANG, J. and J. WU (2017) "A robust sign language recognition system with multiple wi-fi devices," in *Proceedings of the Workshop on Mobility in the Evolving Internet Architecture*, ACM, pp. 19–24.

[48] YANG, K., Y. CHEN, X. CHEN, and W. DU (2023) "Link Quality Modeling for LoRa Networks in Orchards," in *Proceedings of the 22nd International Conference on Information Processing in Sensor Networks*, pp. 27–39.

[49] XU, Y., Y. CHEN, X. ZHANG, X. LIN, P. HU, Y. MA, S. LU, W. DU, Z. MAO, E. ZHAI, ET AL. (2023) "CloudEval-YAML: A Practical Benchmark for Cloud Configuration Generation," *arXiv preprint arXiv:2401.06786*.

[50] XU, Y. ET AL. "CloudEval-YAML: A Realistic and Scalable Benchmark for Cloud Configuration Generation," .

[51] PU, Q., S. GUPTA, S. GOLLAKOTA, and S. PATEL (2013) "Whole-home gesture recognition using wireless signals," in *Proceedings of the 19th annual international conference on Mobile computing & networking*, ACM, pp. 27–38.

[52] REGANI, S. D., C. WU, B. WANG, M. WU, and K. R. LIU (2021) "mmWrite: Passive Handwriting Tracking Using a Single Millimeter Wave Radio," *IEEE Internet of Things Journal*.

[53] KHAMIS, A., B. KUSY, C. T. CHOU, M.-L. MCLAWS, and W. HU (2020) "RFWash: a weakly supervised tracking of hand hygiene technique," in *Proceedings of the 18th Conference on Embedded Networked Sensor Systems*, pp. 572–584.

[54] MA, Y., G. ZHOU, S. WANG, H. ZHAO, and W. JUNG (2018) "Signfi: Sign language recognition using wifi," *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, **2**(1), p. 23.

[55] SANTHALINGAM, P. S., Y. DU, R. WILKERSON, D. ZHANG, P. PATHAK, H. RANGWALA, R. KUSHALNAGAR, ET AL. (2020) "Expressive ASL Recognition using Millimeter-wave Wireless Signals," in *2020 17th Annual IEEE International Conference on Sensing, Communication, and Networking (SECON)*, IEEE, pp. 1–9.

[56] ZHAO, M. ET AL. (2019) "Through-wall human mesh recovery using radio signals," in *IEEE CVPR*.

[57] JIANG, W. ET AL. (2020) "Towards 3D human pose construction using wifi," in *ACM MobiCom*, pp. 1–14.

[58] ZHAO, M. ET AL. (2017) "Learning sleep stages from radio signals: A conditional adversarial architecture," in *ICML*, pp. 4100–4109.

[59] ——— (2016) "Emotion recognition using wireless signals," in *ACM MobiCom*, pp. 95–108.

[60] (2020), "Google Project Soli," `https://atap.google.com/soli/`.

[61] ZHOU, H., T. LU, Y. LIU, S. ZHANG, and M. GOWDA (2022) "Learning on the Rings: Self-Supervised 3D Finger Motion Tracking Using Wearable Sensors," *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, **6**(2), pp. 1–31.

[62] ZHOU, H., T. LU, Y. LIU, S. ZHANG, R. LIU, and M. GOWDA (2023) "One Ring to Rule Them All: An Open Source Smartring Platform for Finger Motion Analytics and Healthcare Applications," in *Proceedings of the 8th ACM/IEEE Conference on Internet of Things Design and Implementation*, pp. 27–38.

[63] WANG, H. ET AL. (2012) "No need to war-drive: Unsupervised indoor localization," in *ACM MobiSys*, pp. 197–210.

[64] PARATE, A. ET AL. (2014) "Risq: Recognizing smoking gestures with inertial sensors on a wristband," in *ACM MobiSys*.

[65] SEN, S. ET AL. (2020) "Annapurna: An automated smartwatch-based eating detection and food journaling system," *Pervasive and Mobile Computing*, p. 101259.

[66] MARIAKAKIS, A. ET AL. (2018) "Drunk user interfaces: Determining blood alcohol level through everyday smartphone tasks," in *CHI Conference on Human Factors in Computing Systems*.

[67] KIM, M. ET AL. (2020) "Golf Swing Segmentation from a Single IMU Using Machine Learning," *Sensors*.

[68] PARK, K.-B. ET AL. (2020) "Deep learning-based smart task assistance in wearable augmented reality," *Robotics and Computer-Integrated Manufacturing*.

[69] ZHANG, H. ET AL. (2020) "PDLens: smartphone knows drug effectiveness among Parkinson's via daily-life activity fusion," in *ACM MobiCom*.

[70] NIRJON, S. ET AL. (2015) "Typingring: A wearable ring platform for text input," in *ACM MobiSys*.

[71] RAURALE, S. ET AL. (2018) "Emg acquisition and hand pose classification for bionic hands from randomly-placed sensors," in *IEEE ICASSP*.

[72] DE SILVA, A. ET AL. (2020) "Real-Time Hand Gesture Recognition Using Temporal Muscle Activation Maps of Multi-Channel sEMG Signals," *arXiv:2002.03159*.

[73] DU, Y. ET AL. (2017) "Semi-Supervised Learning for Surface EMG-based Gesture Recognition." in *IJCAI*.

[74] QUIVIRA, F. ET AL. (2018) "Translating sEMG signals to continuous hand poses using recurrent neural networks," in *2018 IEEE EMBS International Conference on Biomedical & Health Informatics (BHI)*, IEEE.

[75] SOSIN, I. ET AL. (2018) "Continuous gesture recognition from sEMG sensor data with recurrent neural networks and adversarial domain adaptation," in *International Conference on Control, Automation, Robotics and Vision (ICARCV)*, IEEE.

[76] CHEN, X. and Z. J. WANG (2013) "Pattern recognition of number gestures based on a wireless surface EMG system," *Biomedical Signal Processing and Control*, **8**(2), pp. 184–192.

[77] BENALCÁZAR, M. E., C. E. ANCHUNDIA, J. A. ZEA, P. ZAMBRANO, A. G. JARAMILLO, and M. SEGURA (2018) "Real-time hand gesture recognition based on artificial feed-forward neural networks and emg," in *2018 26th European Signal Processing Conference (EUSIPCO)*, IEEE, pp. 1492–1496.

[78] LUO, X. Y., X. Y. WU, L. CHEN, N. HU, Y. ZHANG, Y. ZHAO, L. T. HU, D. D. YANG, and W. S. HOU (2018) "Forearm muscle synergy reducing dimension of the feature matrix in hand gesture recognition," in *2018 3rd International Conference on Advanced Robotics and Mechatronics (ICARM)*, IEEE, pp. 691–696.

[79] SCHABRON, B., Z. ALASHQAR, N. FUHRMAN, K. JIBBE, and J. DESAI (2019) "Artificial Neural Network to Detect Human Hand Gestures for a Robotic Arm Control," in *2019 41st Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, IEEE, pp. 1662–1665.

[80] STAPORNCHAISIT, S., Y. KIM, A. TAKAGI, N. YOSHIMURA, and Y. KOIKE (2019) "Finger Angle estimation from Array EMG system using linear regression model with Independent Component Analysis," *Frontiers in Neurorobotics*, **13**.

[81] PAN, L. ET AL. (2014) "Continuous estimation of finger joint angles under different static wrist motions from sEMG signals," *Biomedical Signal Processing and Control*.

[82] GEORGE, J. A. ET AL. (2020) "Bilaterally mirrored movements improve the accuracy and precision of training data for supervised learning of neural or myoelectric prosthetic control," *arXiv preprint*.

[83] JIANG, N., J. L. VEST-NIELSEN, S. MUCELI, and D. FARINA (2012) "EMG-based simultaneous and proportional estimation of wrist/hand kinematics in uni-lateral transradial amputees," *Journal of neuroengineering and rehabilitation*, **9**(1), pp. 1–11.

[84] LUO, Y. ET AL. (2016) "Text steganography based on ci-poetry generation using Markov chain model," *KSII Transactions on Internet and Information Systems (TIIS)*, **10**(9), pp. 4568–4584.

[85] LUO, Y. and Y. HUANG (2017) "Text steganography with high embedding rate: Using recurrent neural networks to generate chinese classic poetry," in *Proceedings of the 5th ACM workshop on information hiding and multimedia security*, pp. 99–104.

[86] LEE, S. ET AL. (2019) "Intermittent learning: On-device machine learning on intermittently powered system," *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, **3**(4), pp. 1–30.

[87] LUO, Y. ET AL. (2019) "Spoton: Just-in-time active event detection on energy autonomous sensing systems," *Brief Presentations Proceedings (RTAS 2019)*, **9**.

[88] ISLAM, B. ET AL. (2019) "On-device training from sensor data on batteryless platforms," in *Proceedings of the 18th International Conference on Information Processing in Sensor Networks*, pp. 325–326.

[89] LUO, Y. ET AL. (2021) "SmartON: Just-in-time active event detection on energy harvesting systems," in *2021 17th International Conference on Distributed Computing in Sensor Systems (DCOSS)*, IEEE, pp. 35–44.

[90] CIREGAN, D. ET AL. (2012) "Multi-column deep neural networks for image classification," in *CVPR*, IEEE, pp. 3642–3649.

[91] WAN, L. ET AL. (2013) "Regularization of neural networks using dropconnect," in *ICML*, pp. 1058–1066.

[92] SATO, I. ET AL. (2015) "Apac: Augmented pattern classification with neural networks," *arXiv preprint arXiv:1505.03229*.

[93] SIMARD, P. Y. ET AL. (2003) "Best practices for convolutional neural networks applied to visual document analysis." in *Icdar*, vol. 3.

[94] KRIZHEVSKY, A. ET AL. (2012) "Imagenet classification with deep convolutional neural networks," in *Advances in neural information processing systems*.

[95] KANDA, N. ET AL. (2013) "Elastic spectral distortion for low resource speech recognition with deep neural networks," in *2013 IEEE Workshop on Automatic Speech Recognition and Understanding*.

[96] JAITLY, N. ET AL. (2013) "Vocal tract length perturbation improves speech recognition," in *Proc. ICML Workshop on Deep Learning for Audio, Speech and Language*.

[97] UM, T. T. ET AL. (2017) "Data augmentation of wearable sensor data for parkinson's disease monitoring using convolutional neural networks," in *ACM ICMI*.

[98] RASHID, K. M. ET AL. (2019) "Times-series data augmentation and deep learning for construction equipment activity recognition," *Advanced Engineering Informatics*, **42**, p. 100944.

[99] CHANG, Y. ET AL. (2020) "A Systematic Study of Unsupervised Domain Adaptation for Robust Human-Activity Recognition," *ACM IMWUT*.

[100] HUANG, Y. ET AL. (2018) "Deep inertial poser: learning to reconstruct human pose from sparse inertial measurements in real time," *ACM TOG*, **37**(6), pp. 1–15.

[101] (2020), "ViCON - award winning motion capture system," `https://www.vicon.com/`.

[102] REY, V. F. ET AL. (2019) "Let there be IMU data: generating training data for wearable, motion sensor based activity recognition from monocular RGB videos," in *ACM UbiComp/ISWC*, pp. 699–708.

[103] XIAO, F. ET AL. (2020) "A Deep Learning Method for Complex Human Activity Recognition Using Virtual Wearable Sensors," *arXiv preprint arXiv:2003.01874*.

[104] TAKEDA, S. ET AL. (2018) "A multi-sensor setting activity recognition simulation tool," in *ACM UbiComp*, pp. 1444–1448.

[105] KWON, H. ET AL. (2020) "IMUTube: Automatic extraction of virtual on-body accelerometry from video for human activity recognition," *arXiv preprint arXiv:2006.05675*.

[106] ZHANG, S. ET AL. (2020) "Deep generative cross-modal on-body accelerometer data synthesis from videos," in *ACM UbiComp/ISWC*, pp. 223–227.

[107] (2011), "Finger Placement When Shooting a Basketball," `https://www.sportsrec.com/476507-finger-placement-when-shooting-a-basketball.html`.

[108] TRUONG, H. ET AL. (2018) "CapBand: Battery-free Successive Capacitance Sensing Wristband for Hand Gesture Recognition," in *ACM SenSys*.

[109] SIDDIQUI, N. and R. H. CHAN (2020) "Multimodal hand gesture recognition using single IMU and acoustic measurements at wrist," *PloS one*, **15**(1), p. e0227039.

[110] KIM, M., J. CHO, S. LEE, and Y. JUNG (2019) "IMU Sensor-Based Hand Gesture Recognition for Human-Machine Interfaces," *Sensors*, **19**(18), p. 3827.

[111] GEORGI, M., C. AMMA, and T. SCHULTZ (2015) "Recognizing Hand and Finger Gestures with IMU based Motion and EMG based Muscle Activity Sensing." in *Biosignals*, pp. 99–108.

[112] ZHANG, C. ET AL. (2018) "FingerPing: Recognizing Fine-grained Hand Poses using Active Acoustic On-body Sensing," in *ACM CHI*.

[113] NGUYEN, V., S. RUPAVATHARAM, L. LIU, R. HOWARD, and M. GRUTESER (2019) "HandSense: capacitive coupling-based dynamic, micro finger gesture recognition," in *Proceedings of the 17th Conference on Embedded Networked Sensor Systems*, pp. 285–297.

[114] "OSF|SignData.csv," `https://osf.io/ua4mw/`.

[115] (2011), "How many deaf people are there in United States," `https://research.gallaudet.edu/Demographics/deaf-US.php`.

[116] (2020), "World Health Organization: Deafness and Hearing Loss," `https://www.who.int/news-room/fact-sheets/detail/deafness-and-hearing-loss`.

[117] GAŁKA, J., M. MĄSIOR, M. ZABORSKI, and K. BARCZEWSKA (2016) "Inertial motion sensing glove for sign language gesture acquisition and recognition," *IEEE Sensors Journal*, **16**(16), pp. 6310–6316.

[118] KOLLER, O., S. ZARGARAN, H. NEY, and R. BOWDEN (2018) "Deep Sign: Enabling Robust Statistical Continuous Sign Language Recognition via Hybrid CNN-HMMs," *International Journal of Computer Vision*.

[119] HOU, J. ET AL. (2019) "SignSpeaker: A Real-time, High-Precision SmartWatch-based Sign Language Translator," *MobiCom*.

[120] GREFF, K., R. K. SRIVASTAVA, J. KOUTNÍK, B. R. STEUNEBRINK, and J. SCHMIDHUBER (2016) "LSTM: A search space odyssey," *IEEE transactions on neural networks and learning systems*, **28**(10), pp. 2222–2232.

[121] "New Rugged and Compact IMU," `https://variense.com/product/vmu931/`.

[122] TENNANT, R. A. and M. G. BROWN (1998) *The American sign language handshape dictionary*, Gallaudet University Press.

[123] BAHAN, B. J. (1997) "Non-manual realization of agreement in American Sign Language." .

[124] PICHLER, D. C. (2002) "Word order variation and acquisition in American Sign Language." .

[125] GROSJEAN, F. and H. LANE (1977) "Pauses and syntax in American sign language," *Cognition*, **5**(2), pp. 101–117.

[126] "ASL Grammar," https://www.lifeprint.com/asl101/pages-layout/grammar.htm.

[127] "Motiv Ring | 24/7 Smart Ring | Fitness + Sleep Tracking | Online Security," https://mymotiv.com/.

[128] "Oura Ring: The most accurate sleep and activity tracker," https://ouraring.com/.

[129] (2018), "Oura Ring Review," https://www.wareable.com/health-and-wellbeing/oura-ring-2018-review-6628.

[130] (2019), "Oura Ring - What we learned about the sleep tracking ring," https://www.cnbc.com/2019/12/20/oura-ring-review---what-we-learned-about-the-sleep-tracking-ring.html.

[131] (2019), "Oura Ring review - The early adopter catches the worm," https://www.androidauthority.com/oura-ring-2-review-933935/.

[132] SHEN, S. ET AL. (2018) "Closing the Gaps in Inertial Motion Tracking," in *ACM MobiCom*.

[133] COLLOBERT, R., J. WESTON, L. BOTTOU, M. KARLEN, K. KAVUKCUOGLU, and P. KUKSA (2011) "Natural language processing (almost) from scratch," *Journal of machine learning research*, **12**(Aug), pp. 2493–2537.

[134] BERNDT, D. J. and J. CLIFFORD (1994) "Using dynamic time warping to find patterns in time series." in *KDD workshop*, vol. 10, Seattle, WA, pp. 359–370.

[135] SHOURIJEH, M. S., R. S. RAZAVIAN, and J. MCPHEE (2017) "Estimation of Maximum Finger Tapping Frequency Using Musculoskeletal Dynamic Simulations," *Journal of Computational and Nonlinear Dynamics*, **12**(5), p. 051009.

[136] ZHOU, P. ET AL. (2014) "Use it free: Instantly knowing your phone attitude," in *ACM MobiCom*.

[137] VITERBI, A. (1967) "Error bounds for convolutional codes and an asymptotically optimum decoding algorithm," *IEEE transactions on Information Theory*, **13**(2), pp. 260–269.

[138] FANG, B. ET AL. (2017) "DeepASL: Enabling Ubiquitous and Non-Intrusive Word and Sentence-Level Sign Language Translation," in *ACM SenSys*.

[139] BENGIO, Y. and Y. GRANDVALET (2004) "No unbiased estimator of the variance of k-fold cross-validation," *Journal of machine learning research*, **5**(Sep), pp. 1089–1105.

[140] TUNG, Y.-C. and K. G. SHIN (2015) "Echotag: Accurate infrastructure-free indoor location tagging with smartphones," in *ACM MobiCom*.

[141] SHERMAN, M. ET AL. (2014) "User-generated free-form gestures for authentication: Security and memorability," in *ACM MobiSys*.

[142] LIU, J. ET AL. (2009) "uWave: Accelerometer-based personalized gesture recognition and its applications," *Pervasive and Mobile Computing*.

[143] NANDAKUMAR, R. ET AL. (2016) "Fingerio: Using active sonar for fine-grained finger tracking," in *ACM CHI*.

[144] SIDDIQUI, N. and R. H. CHAN (2020) "Multimodal hand gesture recognition using single IMU and acoustic measurements at wrist," *Plos one*, **15**(1), p. e0227039.

[145] TOMASI, C., S. PETROV, and A. SASTRY (2003) "3d tracking= classification+ interpolation," in *null*, IEEE, p. 1441.

[146] BRASHEAR, H., T. STARNER, P. LUKOWICZ, and H. JUNKER (2003) "Using multiple sensors for mobile sign language recognition," Georgia Institute of Technology.

[147] (2017), "Facial expressions in American Sign Language," `https://www.newscientist.com/article/2133451-automatic-sign-language-translators-turn-signing-into-text/`

[148] YE, Y., Y. TIAN, M. HUENERFAUTH, and J. LIU (2018) "Recognizing American Sign Language Gestures from within Continuous Videos," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pp. 2064–2073.

[149] PU, J., W. ZHOU, and H. LI (2019) "Iterative alignment network for continuous sign language recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 4165–4174.

[150] IQBAL, U. ET AL. (2018) "Hand pose estimation via latent 2.5 d heatmap regression," in *ECCV*.

[151] SAGGIO, G. (2012) "Mechanical model of flex sensors used to sense finger movements," *Sensors and Actuators A: Physical*, **185**, pp. 53–58.

[152] SAGGIO, G., F. RIILLO, L. SBERNINI, and L. R. QUITADAMO (2015) "Resistive flex sensors: a survey," *Smart Materials and Structures*, **25**(1), p. 013001.

[153] PRAVEEN, N., N. KARANTH, and M. MEGHA (2014) "Sign language interpreter using a smart glove," in *2014 International Conference on Advances in Electronics Computers and Communications*, IEEE, pp. 1–5.

[154] SHARMA, D., D. VERMA, and P. KHETARPAL (2015) "LabVIEW based Sign Language Trainer cum portable display unit for the speech impaired," in *2015 Annual IEEE India Conference (INDICON)*, IEEE, pp. 1–6.

[155] SHUKOR, A. Z., M. F. MISKON, M. H. JAMALUDDIN, F. BIN ALI, M. F. ASYRAF, M. B. BIN BAHAR, ET AL. (2015) "A new data glove approach for Malaysian sign language detection," *Procedia Computer Science*, **76**, pp. 60–67.

[156] WANG, L., T. MEYDAN, and P. WILLIAMS (2017) "Design and evaluation of a 3-D printed optical sensor for monitoring finger flexion," *IEEE sensors journal*, **17**(6), pp. 1937–1944.

[157] KIM, J.-S., W. JANG, and Z. BIEN (1996) "A dynamic gesture recognition system for the Korean sign language (KSL)," *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, **26**(2), pp. 354–359.

[158] SWEE, T. T., A. ARIFF, S.-H. SALLEH, S. K. SENG, and L. S. HUAT (2007) "Wireless data gloves Malay sign language recognition system," in *2007 6th International Conference on Information, Communications & Signal Processing*, IEEE, pp. 1–4.

[159] JEONG, E., J. LEE, and D. KIM (2011) "Finger-gesture recognition glove using velostat (ICCAS 2011)," in *2011 11th International Conference on Control, Automation and Systems*, IEEE, pp. 206–210.

[160] LEE, H.-K., J. CHUNG, S.-I. CHANG, and E. YOON (2008) "Normal and shear force measurement using a flexible polymer tactile sensor with embedded multiple capacitors," *Journal of Microelectromechanical Systems*, **17**(4), pp. 934–942.

[161] CHOUHAN, T., A. PANSE, A. K. VOONA, and S. SAMEER (2014) "Smart glove with gesture recognition ability for the hearing and speech impaired," in *2014 IEEE Global Humanitarian Technology Conference-South Asia Satellite (GHTC-SAS)*, IEEE, pp. 105–110.

[162] WU, J., L. SUN, and R. JAFARI (2016) "A Wearable System for Recognizing American Sign Language in Real-Time Using IMU and Surface EMG Sensors." *IEEE J. Biomedical and Health Informatics*, **20**(5), pp. 1281–1290.

[163] LI, Y., X. CHEN, X. ZHANG, K. WANG, and Z. J. WANG (2012) "A sign-component-based framework for Chinese sign language recognition using accelerometer and sEMG data," *IEEE transactions on biomedical engineering*, **59**(10), pp. 2695–2704.

[164] (2019), "5DT Data Glove Ultra - 5DT," https://5dt.com/5dt-data-glove-ultra/.

[165] IWASAKO, K., M. SOGA, and H. TAKI (2014) "Development of finger motion skill learning support system based on data gloves," *Procedia Computer Science*, **35**, pp. 1307–1314.

[166] OZ, C. and M. C. LEU (2011) "American Sign Language word recognition with a sensory glove using artificial neural networks," *Engineering Applications of Artificial Intelligence*, **24**(7), pp. 1204–1213.

[167] (2013), "Bracelets and rings translate sign language," https://www.cnet.com/news/bracelet-and-rings-translate-sign-language/.

[168] PARIZI, F. S., E. WHITMIRE, and S. PATEL (2019) "AuraRing: Precise Electromagnetic Finger Tracking," *ACM IMWUT*.

[169] HUANG, J., F. QIAN, A. GERBER, Z. M. MAO, S. SEN, and O. SPATSCHECK (2012) "A close examination of performance and power characteristics of 4G LTE networks," in *ACM MobiSys*.

[170] AGOSTINO, R. ET AL. (2003) "Impairment of individual finger movements in Parkinson's disease," *Movement disorders*, **18**(5), pp. 560–565.

[171] MIKOLOV, T. ET AL. (2010) "Recurrent neural network based language model," in *Eleventh annual conference of the international speech communication association*.

[172] MIKOLOV, T., K. CHEN, G. S. CORRADO, and J. A. DEAN (2015), "Computing numeric representations of words in a high-dimensional space," US Patent 9,037,464.

[173] MILLER, G. A. (1995) "WordNet: a lexical database for English," *Communications of the ACM*, **38**(11), pp. 39–41.

[174] BAKER, C. F., C. J. FILLMORE, and J. B. LOWE (1998) "The berkeley framenet project," in *Proceedings of the 17th international conference on Computational linguistics-Volume 1*.

[175] ANDO, T., Y. KUBO, B. SHIZUKI, and S. TAKAHASHI (2017) "CanalSense: face-related movement recognition system based on sensing air pressure in ear canals," in *ACM UIST*.

[176] IRAVANTCHI ET AL. (2019) "Interferi: Gesture Sensing using On-Body Acoustic Interferometry," in *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*, ACM, p. 276.

[177] NGUYEN, A. ET AL. (2016) "A lightweight, inexpensive in-ear sensing system for automatic whole-night sleep stage monitoring," in *ACM SenSys*.

[178] (2021), "Augmented reality for the web," https://developers.google.com/web/updates/2018/06/ar-for-the-web.

[179] LAYONA, R. ET AL. (2018) "Web based augmented reality for human body anatomy learning," *Procedia Computer Science*.

[180] (2021), "Web XR Device API," https://www.w3.org/TR/webxr/.

[181] ZHANG, Y. ET AL. (2015) "Tomo: Wearable, low-cost electrical impedance tomography for hand gesture recognition," in *ACM UIST*.

[182] MCINTOSH, J. ET AL. (2017) "Echoflex: Hand gesture recognition using ultrasound imaging," in *2017 CHI Conference on Human Factors in Computing Systems*.

[183] DEMENTYEV, A. and J. A. PARADISO (2014) "WristFlex: low-power gesture input with wrist-worn pressure sensors," in *ACM UIST*.

[184] KIM, D. ET AL. (2012) "Digits: freehand 3D interactions anywhere using a wrist-worn gloveless sensor," in *ACM UIST*.

[185] GIESER, S. N. ET AL. (2017) "Evaluation of a low cost emg sensor as a modality for use in virtual reality applications," in *International Conference on Virtual, Augmented and Mixed Reality*, Springer.

[186] XU, S. ET AL. (2019) "The Effectiveness of Virtual Reality in Safety Training: Measurement of Emotional Arousal with Electromyography," in *ISARC*.

[187] "Myo resurrected? Facebook acquires CTRL-labs in device gesture-control research push," https://www.zdnet.com/article/facebook-acquires-ctrl-labs-in-machine-mind-control-research-push/.

[188] "Facebook Might Have Just Given Us a Peek at Our Wild AR Future," https://www.gizmodo.com.au/2020/09/facebook-might-have-just-given-us-a-peek-at-our-wild-ar-future/.

[189] BADRINARAYANAN, V., A. KENDALL, and R. CIPOLLA (2017) "Segnet: A deep convolutional encoder-decoder architecture for image segmentation," *IEEE transactions on pattern analysis and machine intelligence*, pp. 2481–2495.

[190] HE, K. ET AL. (2016) "Deep residual learning for image recognition," in *IEEE CVPR*.

[191] STASHUK, D. (2001) "EMG signal decomposition: how can it be accomplished and used?" *Journal of Electromyography and Kinesiology*, **11**(3), pp. 151–173.

[192] KOSHIO, T. ET AL. (2012) "Identification of surface and deep layer muscles activity by surface EMG," in *2012 Proceedings of SICE Annual Conference*, IEEE.

[193] WINKEL, J. ET AL. (1991) "Significance of skin temperature changes in surface electromyography," *European journal of applied physiology and occupational physiology*.

[194] "Myo official tutorial," https://support.getmyo.com/hc/en-us/articles/203910089-Warm-up-while-wearing-your-Myo-armband.

[195] "Smart skin: Electronics that stick and stretch like a temporary tattoo," https://www.vice.com/en_us/article/nee8qm/this-tattoo-can-monitor-your-heart-rate-and-brain-waves.

[196] POLYGERINOS, P. ET AL. (2015) "EMG controlled soft robotic glove for assistance during activities of daily living," in *2015 IEEE international conference on rehabilitation robotics (ICORR)*, IEEE.

[197] LAWRENCE, S. ET AL. (1997) "Face recognition: A convolutional neural-network approach," *IEEE transactions on neural networks*.

[198] IOFFE, S. ET AL. (2015) "Batch normalization: Accelerating deep network training by reducing internal covariate shift," *arXiv preprint arXiv:1502.03167*.

[199] ARORA, R. ET AL. (2016) "Understanding deep neural networks with rectified linear units," *arXiv preprint arXiv:1611.01491*.

[200] WAGER, S., S. WANG, and P. S. LIANG (2013) "Dropout training as adaptive regularization," in *Advances in neural information processing systems*, pp. 351–359.

[201] HASAN, S. and C. A. LINTE (2019) "U-NetPlus: a modified encoder-decoder U-Net architecture for semantic and instance segmentation of surgical instrument," *arXiv preprint arXiv:1902.08994*.

[202] LIEW, S. S. ET AL. (2016) "Bounded activation functions for training stability of deep neural networks on visual pattern recognition problems," *Neurocomputing*.

[203] KRIZHEVSKY, A. ET AL. (2012) "Imagenet classification with deep convolutional neural networks," in *NIPS*.

[204] DENG, J. ET AL. (2009) "Imagenet: A large-scale hierarchical image database," in *IEEE CVPR*.

[205] ZHOU, Z. ET AL. (2017) "Fine-tuning convolutional neural networks for biomedical image analysis: actively and incrementally," in *IEEE CVPR*.

[206] HAN, X. ET AL. (2017) "Pre-trained alexnet architecture with pyramid pooling and supervision for high spatial resolution remote sensing image scene classification," *Remote Sensing*.

[207] NAWAZ, W. ET AL. (2018) "Classification of breast cancer histology images using alexnet," in *International conference image analysis and recognition*, Springer.

[208] DEVLIN, J. ET AL. (2018) "Bert: Pre-training of deep bidirectional transformers for language understanding," *arXiv preprint arXiv:1810.04805*.

[209] ZHANG, H., J. XU, and J. WANG (2019) "Pretraining-based natural language generation for text summarization," *arXiv preprint arXiv:1902.09243*.

[210] QU, C. ET AL. (2019) "BERT with history answer embedding for conversational question answering," in *ACM SIGIR Conference on Research and Development in Information Retrieval*.

[211] CHANG, W.-G. ET AL. (2019) "Domain-specific batch normalization for unsupervised domain adaptation," in *IEEE CVPR*.

[212] MANCINI, M. ET AL. (2018) "Boosting domain adaptation by discovering latent domains," in *IEEE CVPR*.

[213] PASCANU, R., T. MIKOLOV, and Y. BENGIO (2012) "Understanding the exploding gradient problem," *CoRR, abs/1211.5063*, **2**.

[214] JAEGER, H. (2002) *Tutorial on training recurrent neural networks, covering BPPT, RTRL, EKF and the" echo state network" approach*, vol. 5, GMD-Forschungszentrum Informationstechnik Bonn.

[215] ABADI, M. ET AL. (2016) "Tensorflow: A system for large-scale machine learning," in *OSDI*.

[216] KINGMA, D. P. and J. BA (2014) "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*.

[217] BERTERO, M., C. DE MOL, and G. A. VIANO (1980) "The stability of inverse problems," in *Inverse scattering problems in optics*, Springer, pp. 161–214.

[218] "Myo warmup," `https://support.getmyo.com/hc/en-us/articles/203910089-Warm-up-while-wearing-your-Myo-armband`.

[219] "powerconsump," `https://pdfs.semanticscholar.org/6c0c/af5d51def3730bb746535099252b724ddd31.pdf`.

[220] "Profile battery usage with Batterystats and Battery Historian," `https://developer.android.com/topic/performance/power/setup-battery-historian`.

[221] WANG, J. ET AL. (2014) "Ubiquitous keyboard for mobile devices: harnessing multipath fading for fine-grained keystroke localization," in *ACM MobiCom*.

[222] XIONG, J. and K. JAMIESON (2013) "Arraytrack: A fine-grained indoor location system," in *USENIX NSDI*.

[223] LIU, Y. ET AL. (2020) "Finger Gesture Tracking for Interactive Applications: A Pilot Study with Sign Languages," *ACM IMWUT*.

[224] ——— (2020) "Application Informed Motion Signal Processing for Finger Motion Tracking Using Wearable Sensors," in *IEEE ICASSP*.

[225] TZENG, E. ET AL. (2017) "Adversarial discriminative domain adaptation," in *CVPR*.

[226] SUN, B. and K. SAENKO (2016) "Deep coral: Correlation alignment for deep domain adaptation," in *European conference on computer vision*, Springer.

[227] MAUDRICH, T., R. KENVILLE, J. LEPSIEN, A. VILLRINGER, P. RAGERT, and C. J. STEELE (2017) "Mirror electromyografic activity in the upper and lower extremity: a comparison between endurance athletes and non-athletes," *Frontiers in human neuroscience*, **11**, p. 485.

[228] UTTNER, I., E. KRAFT, D. A. NOWAK, F. MÜLLER, J. PHILIPP, A. ZIERDT, and J. HERMSDÖRFER (2007) "Mirror movements and the role of handedness: isometric grip forces changes." *Motor control*, **11**(1).

[229] (2021), "Finger and Partial Hand Prosthetic Options," `https://www.armdynamics.com/our-care/finger-and-partial-hand-prosthetic-options`.

[230] HUININK, L. H., H. BOUWSEMA, D. H. PLETTENBURG, C. K. VAN DER SLUIS, and R. M. BONGERS (2016) "Learning to use a body-powered prosthesis: changes in functionality and kinematics," *Journal of neuroengineering and rehabilitation*, **13**(1), pp. 1–12.

[231] CAREY, S. L., D. J. LURA, and M. J. HIGHSMITH (2015) "Differences in myoelectric and body-powered upper-limb prostheses: Systematic literature review." *Journal of Rehabilitation Research & Development*, **52**(3).

[232] DAVIS, T. S., H. A. WARK, D. HUTCHINSON, D. J. WARREN, K. O'NEILL, T. SCHEINBLUM, G. A. CLARK, R. A. NORMANN, and B. GREGER (2016) "Restoring motor control and sensory feedback in people with upper extremity amputations using arrays of 96 microelectrodes implanted in the median and ulnar nerves," *Journal of neural engineering*, **13**(3), p. 036001.

[233] HESSE, S., H. KUHLMANN, J. WILK, C. TOMELLERI, and S. G. KIRKER (2008) "A new electromechanical trainer for sensorimotor rehabilitation of paralysed fingers: a case series in chronic and acute stroke patients," *Journal of neuroengineering and rehabilitation*, **5**(1), p. 21.

[234] BARBOSA, P. ET AL. (2018) "Unsupervised domain adaptation for human activity recognition," in *International Conference on Intelligent Data Engineering and Automated Learning*, Springer, pp. 623–630.

[235] XIANG, D. ET AL. (2019) "Monocular total capture: Posing face, body, and hands in the wild," in *IEEE CVPR*.

[236] CAO, Z. ET AL. (2018) "OpenPose: realtime multi-person 2D pose estimation using Part Affinity Fields," *arXiv preprint arXiv:1812.08008*.

[237] "Wearables For Motion Tracking + Wireless Environment Monitoring," `https://mbientlab.com/store/adhesive-sensor-research-kit/`.

[238] "Signing Savvy – ASL Sign Language Video Dictionary," `https://www.signingsavvy.com/`.

[239] IONESCU, C. ET AL. (2013) "Human3. 6m: Large scale datasets and predictive methods for 3d human sensing in natural environments," *IEEE transactions on pattern analysis and machine intelligence*, **36**(7), pp. 1325–1339.

[240] BACHLIN, M. ET AL. (2009) "Wearable assistant for Parkinson's disease patients with the freezing of gait symptom," *IEEE Transactions on Information Technology in Biomedicine*, **14**(2), pp. 436–446.

[241] REISS, A. ET AL. (2012) "Introducing a new benchmarked dataset for activity monitoring," in *IEEE ISWC*, pp. 108–109.

[242] SHAEFFER, D. K. (2013) "MEMS inertial sensors: A tutorial overview," *IEEE Communications Magazine*, **51**(4), pp. 100–109.

[243] ROY, N. ET AL. (2014) "I am a smartphone and i can tell my user's walking direction," in *ACM MobiSys*.

[244] SHEN, S. ET AL. (2016) "I am a Smartwatch and I can Track my User's Arm," in *ACM MobiCom*.

[245] YOUNG, A. D. ET AL. (2011) "IMUSim: A simulation environment for inertial sensing algorithm design and evaluation," in *ACM/IEEE IPSN*.

[246] LIU, Y. ET AL. "Finger Gesture Tracking for Interactive Applications: A Pilot Study with Sign Languages," *ACM IMWUT 2020*.

[247] ZHANG, L. ET AL. (2022) "Demo Abstract: Capuchin: A Neural Network Model Generator for 16-bit Microcontrollers," in *2022 21st ACM/IEEE International Conference on Information Processing in Sensor Networks (IPSN)*, IEEE, pp. 497–498.

[248] QI, Y. ET AL. (2022) "Blockchain-Based Light-Weighted Provable Data Possession for Low Performance Devices." *Computers, Materials & Continua*, **73**(2).

[249] MONJUR, M. ET AL. (2023) "SoundSieve: Seconds-Long Audio Event Recognition on Intermittently-Powered Systems," in *Proceedings of the 21st Annual International Conference on Mobile Systems, Applications and Services*, pp. 28–41.

[250] QI, Y. ET AL. (2023) "Blockchain-Based Privacy-Preserving Public Auditing for Group Shared Data." *Intelligent Automation & Soft Computing*, **35**(3).

[251] SOCHER, R. ET AL. (2013) "Zero-shot learning through cross-modal transfer," in *NeurIPS*, pp. 935–943.

[252] CUBUK, E. D., B. ZOPH, D. MANE, V. VASUDEVAN, and Q. V. LE (2019) "Autoaugment: Learning augmentation strategies from data," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 113–123.

[253] KONG, H., X. XU, J. YU, Q. CHEN, C. MA, Y. CHEN, Y.-C. CHEN, and L. KONG (2022) "m3Track: mmwave-based multi-user 3D posture tracking," in *Proceedings of the 20th Annual International Conference on Mobile Systems, Applications and Services*, pp. 491–503.

[254] CHANG, Z., F. ZHANG, J. XIONG, J. MA, B. JIN, and D. ZHANG (2022) "Sensor-free Soil Moisture Sensing Using LoRa Signals," *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, **6**(2), pp. 1–27.

[255] HA, U., J. LENG, A. KHADDAJ, and F. ADIB (2020) "Food and Liquid Sensing in Practical Environments using {RFIDs}," in *17th USENIX Symposium on Networked Systems Design and Implementation (NSDI 20)*, pp. 1083–1100.

[256] HA, U., S. MADANI, and F. ADIB (2021) "WiStress: Contactless stress monitoring using wireless signals," *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, **5**(3), pp. 1–37.

[257] AN, S. and U. Y. OGRAS (2021) "MARS: mmWave-based Assistive Rehabilitation System for Smart Healthcare," *ACM Transactions on Embedded Computing Systems (TECS)*, **20**(5s), pp. 1–22.

[258] ZHAO, Y., V. SARK, M. KRSTIC, and E. GRASS (2022) "Novel Approach for Gesture Recognition Using mmWave FMCW RADAR," in *2022 IEEE 95th Vehicular Technology Conference:(VTC2022-Spring)*, IEEE, pp. 1–6.

[259] BANSAL, K., K. RUNGTA, S. ZHU, and D. BHARADIA (2020) "Pointillism: Accurate 3d bounding box estimation with multi-radars," in *Proceedings of the 18th Conference on Embedded Networked Sensor Systems*, pp. 340–353.

[260] SONG, J., G. SÖRÖS, F. PECE, S. R. FANELLO, S. IZADI, C. KESKIN, and O. HILLIGES (2014) "In-air gestures around unmodified mobile devices," in *Proceedings of the 27th annual ACM symposium on User interface software and technology*, pp. 319–329.

[261] KIM, D., O. HILLIGES, S. IZADI, A. D. BUTLER, J. CHEN, I. OIKONOMIDIS, and P. OLIVIER (2012) "Digits: freehand 3D interactions anywhere using a wrist-worn gloveless sensor," in *Proceedings of the 25th annual ACM symposium on User interface software and technology*, pp. 167–176.

[262] ZHAI, S., P. MILGRAM, and W. BUXTON (1996) "The influence of muscle groups on performance of multiple degree-of-freedom input," in *Proceedings of the SIGCHI conference on Human factors in computing systems*, pp. 308–315.

[263] CHEN, A. T.-Y., M. BIGLARI-ABHARI, and K. I.-K. WANG (2018) "Context is King: Privacy Perceptions of Camera-based Surveillance," in *2018 15th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS)*, pp. 1–6.

[264] JIANG, C., J. GUO, Y. HE, M. JIN, S. LI, and Y. LIU (2020) "mmVib: micrometer-level vibration measurement with mmwave radar," in *Proceedings of the 26th Annual International Conference on Mobile Computing and Networking*, pp. 1–13.

[265] QIAN, K., Z. HE, and X. ZHANG (2020) "3D point cloud generation with millimeter-wave radar," *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, **4**(4), pp. 1–23.

[266] BANSAL, K., K. RUNGTA, S. ZHU, and D. BHARADIA (2020) "Pointillism: accurate 3D bounding box estimation with multi-radars," in *Proceedings of the 18th Conference on Embedded Networked Sensor Systems*, pp. 340–353.

[267] SANTHALINGAM, P. S., A. A. HOSAIN, D. ZHANG, P. PATHAK, H. RANGWALA, and R. KUSHALNAGAR (2020) "mmASL: Environment-Independent ASL Gesture Recognition Using 60 GHz Millimeter-wave Signals," *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, **4**(1), pp. 1–30.

[268] LIU, H., Y. WANG, A. ZHOU, H. HE, W. WANG, K. WANG, P. PAN, Y. LU, L. LIU, and H. MA (2020) "Real-time arm gesture recognition in smart home scenarios via millimeter wave sensing," *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, **4**(4), pp. 1–28.

[269] "IWR6843ISK," Https://www.ti.com/tool/IWR6843ISK.

[270] RAO, S. (2017) "Introduction to mmWave sensing: FMCW radars," *Texas Instruments (TI) mmWave Training Series*.

[271] "WaveFarer Radar Simulation Software," `https://www.remcom.com/wavefarer-automotive-radar-software`.

[272] LING, H., R.-C. CHOU, and S.-W. LEE (1989) "Shooting and bouncing rays: Calculating the RCS of an arbitrarily shaped cavity," *IEEE Transactions on Antennas and propagation*, **37**(2), pp. 194–205.

[273] KIM, J. H., H. J. CHUN, I. P. HONG, Y. J. KIM, and Y. B. PARK (2014) "Analysis of FSS radomes based on physical optics method and ray tracing technique," *IEEE Antennas and Wireless Propagation Letters*, **13**, pp. 868–871.

[274] MICHAELI, A. (1984) "Equivalent edge currents for arbitrary aspects of observation," *IEEE Transactions on Antennas and Propagation*, **32**(3), pp. 252–258.

[275] KOUYOUMJIAN, R. G. and P. H. PATHAK (1974) "A uniform geometrical theory of diffraction for an edge in a perfectly conducting surface," *Proceedings of the IEEE*, **62**(11), pp. 1448–1461.

[276] SKIDMORE, G., T. CHAWLA, and G. BEDROSIAN (2019) "Combining Physical Optics and Method of Equivalent Currents to Create Unique Near-Field Propagation and Scattering Technique for Automotive Radar Applications," in *2019 IEEE International Conference on Microwaves, Antennas, Communications and Electronic Systems (COMCAS)*, IEEE, pp. 1–6.

[277] "Using Wavefarer Automotive Radar Simulation Software And Chirp Doppler To Assess Radar Performance For Drive Scenario," https://resources.remcom.com/automotive-radar/publications-wavefarer-chirp-doppler-for-drive-scenarios-comcas201

[278] LI, Y., N. WANG, J. SHI, X. HOU, and J. LIU (2018) "Adaptive batch normalization for practical domain adaptation," *Pattern Recognition*, **80**, pp. 109–117.

[279] LI, Y., N. WANG, J. SHI, J. LIU, and X. HOU (2016) "Revisiting batch normalization for practical domain adaptation," *arXiv preprint arXiv:1603.04779*.

[280] PENG, Y., S. YAN, and Z. LU (2019) "Transfer learning in biomedical natural language processing: an evaluation of BERT and ELMo on ten benchmarking datasets," *arXiv preprint arXiv:1906.05474*.

[281] "patch camelyon Tensorflow datasets," https://www.tensorflow.org/datasets/catalog/patch_camelyon.

[282] KANDEL, I. and M. CASTELLI (2020) "How deeply to fine-tune a convolutional neural network: a case study using a histopathology dataset," *Applied Sciences*, **10**(10), p. 3359.

[283] "Real-time data-capture adapter for radar sensing evaluation module," https://www.ti.com/tool/DCA1000EVM.

[284] "TI mmWave radar," https://dev.ti.com/tirex/explore/node.

[285] LIN, J. and T. S. H. WU "Modeling the Constraints of Human Hand Motion," .

[286] TOMASI, C., S. PETROV, and A. SASTRY (2003) "3D Tracking= Classification+ Interpolation." in *ICCV*, vol. 3, p. 1441.

[287] CAO, Z., I. RADOSAVOVIC, A. KANAZAWA, and J. MALIK (2021) "Reconstructing hand-object interactions in the wild," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 12417–12426.

[288] "Hanes Men's Full-Zip Eco-Smart Hoodie," https://www.amazon.com/Hanes-EcoSmart-Fleece-Hoodie-Black/dp/B00JUM4CT4/.

[289] "YASRKML 3 Panel Room Divider," https://www.amazon.com/YASRKML-Partition-Separators-Freestanding-102x71-3/dp/B092HQ5W7D/.

[290] "American manual alphabet," https://en.wikipedia.org/wiki/American_manual_alphabet.

[291] WANG, W., A. X. LIU, and K. SUN (2016) "Device-Free Gesture Tracking Using Acoustic Signals," Association for Computing Machinery, New York, NY, USA. URL https://doi.org/10.1145/2973750.2973764

[292] MUELLER, F., F. BERNARD, O. SOTNYCHENKO, D. MEHTA, S. SRIDHAR, D. CASAS, and C. THEOBALT (2018) "GANerated Hands for Real-Time 3D Hand Tracking from Monocular RGB," in *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 49–59.

[293] ZHANG, F., V. BAZAREVSKY, A. VAKUNOV, A. TKACHENKA, G. SUNG, C.-L. CHANG, and M. GRUNDMANN (2020) "Mediapipe hands: On-device real-time hand tracking," *arXiv preprint arXiv:2006.10214*.

[294] WU, E., Y. YUAN, H.-S. YEO, A. QUIGLEY, H. KOIKE, and K. M. KITANI (2020) "Back-Hand-Pose: 3D Hand Pose Estimation for a Wrist-worn Camera via Dorsum Deformation Network," in *Proceedings of the 33rd Annual ACM Symposium on User Interface Software and Technology*, pp. 1147–1160.

[295] ZHANG, S., Y. LIU, and M. GOWDA (2023) "I Spy You: Eavesdropping Continuous Speech on Smartphones via Motion Sensors," *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, **6**(4), pp. 1–31.

[296] ZHANG, S. ET AL. (2022) "Let's Grab a Drink: Teacher-Student Learning for Fluid Intake Monitoring using Smart Earphones," in *2022 IEEE/ACM Seventh International Conference on Internet-of-Things Design and Implementation (IoTDI)*, IEEE, pp. 55–66.

[297] ZHANG, S., T. LU, H. ZHOU, Y. LIU, R. LIU, and M. GOWDA (2023) "I Am an Earphone and I Can Hear My User's Face: Facial Landmark Tracking Using Smart Earphones," *ACM Transactions on Internet of Things*, **5**(1), pp. 1–29.

[298] XIA, Z., Y. LUOMEI, C. ZHOU, and F. XU (2020) "Multidimensional feature representation and learning for robust hand-gesture recognition on commercial millimeter-wave radar," *IEEE Transactions on Geoscience and Remote Sensing*, **59**(6), pp. 4749–4764.

[299] REN, Y., J. LU, A. BELETCHI, Y. HUANG, I. KARMANOV, D. FONTIJNE, C. PATEL, and H. XU (2021) "Hand gesture recognition using 802.11 ad mmWave sensor in the mobile device," in *2021 IEEE Wireless Communications and Networking Conference Workshops (WCNCW)*, IEEE, pp. 1–6.

[300] ZHANG, Z., Z. TIAN, M. ZHOU, W. NIE, and Z. LI (2018) "Riddle: Real-time interacting with hand description via millimeter-wave sensor," in *2018 IEEE International Conference on Communications (ICC)*, IEEE, pp. 1–6.

[301] CHINTALAPUDI, K. ET AL. (2010) "Indoor localization without the pain," in *ACM MobiCom*.

[302] ZHANG, Y. and C. HARRISON (2015) "Tomo: Wearable, low-cost electrical impedance tomography for hand gesture recognition," in *Proceedings of the 28th Annual ACM Symposium on User Interface Software & Technology*, pp. 167–173.

[303] IRAVANTCHI, Y., Y. ZHANG, E. BERNITSAS, M. GOEL, and C. HARRISON (2019) "Interferi: Gesture sensing using on-body acoustic interferometry," in *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*, pp. 1–13.

[304] KIENZLE, W., E. WHITMIRE, C. RITTALER, and H. BENKO (2021) "ElectroRing: Subtle Pinch and Touch Detection with a Ring," in *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems*, pp. 1–12.

[305] FANG, B., J. CO, and M. ZHANG (2017) "Deepasl: Enabling ubiquitous and non-intrusive word and sentence-level sign language translation," in *Proceedings of the 15th ACM Conference on Embedded Network Sensor Systems*, pp. 1–13.

[306] SUN, W., F. M. LI, C. HUANG, Z. LEI, B. STEEPER, S. TAO, F. TIAN, and C. ZHANG (2021) "ThumbTrak: Recognizing Micro-finger Poses Using a Ring with Proximity Sensing," *arXiv preprint arXiv:2105.14680*.

[307] LIU, Y., C. LIN, and Z. LI (2021) "WR-Hand: Wearable Armband Can Track User's Hand," *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, **5**(3), pp. 1–27.

[308] TORRES, T. (2015), "Myo Gesture Control Armband Review," `https://www.pcmag.com/reviews/myo-gesture-control-armband`.

[309] MITCHELL, R. E. (2005) "How many deaf people are there in the United States? Estimates from the Survey of Income and Program Participation," *Journal of deaf studies and deaf education*, **11**(1), pp. 112–119.

[310] DU, G., P. ZHANG, J. MAI, and Z. LI (2012) "Markerless kinect-based hand tracking for robot teleoperation," *International Journal of Advanced Robotic Systems*, **9**(2), p. 36.

[311] ASHWINI, A. (2020), "Everything You Need To Know About IoT Prototyping," Https://medium.com/swlh/everything-you-need-to-know-about-iot-prototyping-e4ad2739bc6a.

Vita

Yilin Liu

Yilin Liu was born in Sichuan, southwest of China, in 1997. As the only child, Yilin grew up in a family where education was highly valued, and he developed a passion for technology from a young age. Yilin's academic journey began at the School of the Gifted Young, University of Science and Technology of China (USTC) when he was only 16, where he pursued a Bachelor's degree in Electronic Information Science and Technology. In 2018, Yilin joined Pennsylvania State University (PSU) as a Ph.D. candidate in Computer Science Engineering. During his doctoral study under the guidance of Mahanth Gowda, Yilin's research focused on human behaviour sensing, wearables and machine learning. His work has led to significant contributions, as evidenced by publications in WWW, IMWUT and IoTDI. Apart from academic pursuits, Yilin gained practical experience through internships at renowned companies. He interned at Google, Snap Inc. and NEC Laboratories America Inc. Post-graduation, Yilin aspires to join Google to continue his research on wearable sensing.