Title

Enhancing Accessible Communication: Assistive AR System in Bridging the Deaf and Hearing Divide

Author

Guo, Yunqi

Publication Date

2023

Peer reviewed|Thesis/dissertation

UNIVERSITY OF CALIFORNIA
Los Angeles

Enhancing Accessible Communication:
Assistive AR System in Bridging the Deaf and Hearing Divide

A dissertation submitted in partial satisfaction
of the requirements for the degree
Doctor of Philosophy in Computer Science

by

Yunqi Guo

2023

ABSTRACT OF THE DISSERTATION

Enhancing Accessible Communication:
Assistive AR System in Bridging the Deaf and Hearing Divide

by

Yunqi Guo
Doctor of Philosophy in Computer Science
University of California, Los Angeles, 2023
Professor Songwu Lu, Chair

Communication barriers between deaf and hearing individuals have led to difficulties in various real-world scenarios, including emergencies, online meetings, and daily interactions. Mobile and Augmented Reality (AR) systems hold promises for offering low-cost solutions for accessible communication. This dissertation aims to bridge the gap between sign language and other oral language users by leveraging assistive Augmented Reality (AR) glasses for everyday wear. We seek to address the full spectrum issues of sign capture, recognition, and language translation on AR glasses and mobile devices.

Our approach first prioritizes the most urgent setting, emergency communication, before extending to more general settings such as daily, online, and learning interactions. We employ domain-specific models derived from sign language with three main components: 1) Capturing sign gestures on a mobile-glass setup using sign parameters; 2) Recognizing signs with lightweight models based on sign parameter correlations and constraints; and 3) Providing general bidirectional ASL-English translation using ASL grammar and word order correlation.

In collaboration with ASL users, our evaluation results demonstrate that, starting from emergency communication scenarios, our domain-oriented models substantially reduce latency by one to two orders of magnitude compared to previous solutions, while maintaining accurate translations. Furthermore, the developed mobile and AR platforms enable sign language interactions across various settings, such as daily in-person communication, virtual communication, and sign language learning, thereby extending the system's applicability. Our research offers an innovative approach to promote accessible communication, fostering social inclusion, and minimizing communication-related inequalities between deaf and hearing individuals.

The dissertation of Yunqi Guo is approved.

Omid Salehi-Abari

Yingnian Wu

Lixia Zhang

Songwu Lu, Committee Chair

University of California, Los Angeles

2023

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGMENTS

I am deeply grateful to my advisor, Prof. Songwu Lu, for his guidance, support, and encouragement throughout my Ph.D. journey. His intellectual curiosity, passion for research, and commitment to academic excellence have been truly inspiring, and his mentorship has been instrumental in shaping my growth as a researcher. I am honored to have had the opportunity to learn from him and work together. Simultaneously, he encourages me to push my boundaries and fosters stimulating conversations when I encounter obstacles. The assistance and encouragement have been indispensable, and I will remain forever grateful for the guidance.

I am grateful to Professor Lixia Zhang, Professor Yingnian Wu, and Professor Omid Abari, who served as esteemed members of my Ph.D. dissertation committee. Their invaluable feedback, constructive criticism, and expert guidance have significantly refined my research and broadened my intellectual horizons.

My sincere appreciation goes to the Computer Science admission committee and the Computer Science Department at UCLA, the harbor from which my academic voyage set sail. I am also thankful for the financial support I received through the Graduate Student Researcher Fellowship and Teaching Fellowship, which were instrumental in fueling my academic journey.

A special note of thanks goes to my colleagues and friends at the WiNG Research Lab. The camaraderie, intellectual assistance, and unwavering support of Prof. Yuanjie Li, Prof. Chunyi Peng, Prof. Muhammad Taqi Raza, Dr. Zhaowei Tan, Dr. Qianru Li, Dr. Zhehui Zhang, Dr. Jinghao Zhao, Zengwen Yuan, Boyan Ding, Yifei Xu, and Yunze Long have made this expedition both rewarding and enriching.

I am particularly grateful for the unique perspectives and expertise provided by the UCLA HandsOn Club and the Deaf community. My understanding of Deaf culture has been deepened thanks to Prof. Benjamin Lewis, Jennifer Miyaki, Matthew Issa Aboudi,

and Mark-Anthony Valentín. I am particularly grateful for the passion and vision that my team on the AnySign project brought to the table. Their unwavering dedication to building a meaningful project that serves the community has been inspiring. Collaboration with Congkai Tan, Weichong Ling, Yuefu Liu, Ethan Huang, and others has given me valuable insights and has been a highlight of my Ph.D. journey.

I am grateful for the guidance and expertise provided by Petros Zerfos and Xuan-hong Dang from IBM Research in machine learning systems. I also extend my appreciation to Prof. Demetri Terzopoulos and Prof. Asish Law for sparking my interest in computer graphics and AR. The distinguished professors from SJTU, including Prof. Xinbing Wang, Prof. Yong Yu, and Prof. Kai Yu provided invaluable guidance in foundational engineering principles during my undergraduate studies, which has been pivotal in sustaining my academic journey.

I would like to express my gratitude to Sonia for coming up with the name "Assistive AR" and for providing consistent support throughout my Ph.D. years. I would also like to thank Xiaoyang Liu, Guangxuan Xu, and all of my friends who have provided unwavering friendship and constant motivation during this time. Your support has been a source of great strength and has made my journey all the more rewarding.

Finally, I reserve my deepest gratitude to my family. Their faith in my abilities and unwavering encouragement have been my lodestar, inspiring me to use my work for the betterment of society. Their enduring love and support have been my constant pillars of strength.

VITA

2012–2016	B.S., Computer Science & Engineering, SJTU, Shanghai, China
2016–2018	M.S., Computer Science, UCLA.
2018–2023	Graduate Student Researcher, UCLA.
2019–2023	Teaching Assistant, Computer Science, UCLA.
2019&2020 Summer	Research Intern, IBM T.J. Waston Research Center, Yorktown Heights, NY.

PUBLICATIONS

Yunqi Guo, Jinghao Zhao, Boyan Ding, Congtai Tan, Weichong Ling, Zhaowei Tan, Jennifer Miyaki, Hongzhe Du, Songwu Lu. Sign-to-911: Emergency Call Service for Sign Language Users with Assistive AR Glasses, ACM MobiCom 2023.

Yunqi Guo* Zhaowei Tan*, Kaiyuan Chen, Songwu Lu, Ying Nian Wu. A Model Obfuscation Approach to IoT Security, IEEE CNS 2021 (*Co-first author)

Zhaowei Tan, Jinghao Zhao, Yuanjie Li, Yifei Xu, **Yunqi Guo**, Songwu Lu. LDRP: Device-Centric Latency Diagnostic and Reduction for Cellular Networks without Root, IEEE TMC 2023

Zhaowei Tan, Boyan Ding, Jinghao Zhao, **Yunqi Guo**, Songwu Lu. Breaking Cellular IoT with Forged Data-Plane Signaling: Attacks and Countermeasure, ACM TOSN 2022

Muhammad Taqi Raza, **Yunqi Guo**, Songwu Lu, Fatima Muhammad Anwar. On Key Reinstallation Attacks over 4G LTE Control-Plane: Feasibility and Negative Impact, ACM ACSAC 2021

Jinghao Zhao, Boyan Ding, **Yunqi Guo**, Zhaowei Tan, Songwu Lu. SecureSIM: Rethinking Authentication and Access Control for SIM/eSIM, AMC MobiCom 2021

Zhaowei Tan, Boyan Ding, Jinghao Zhao, **Yunqi Guo**, Songwu Lu. Data-Plane Signaling in Cellular IoT: Attacks and Defense, ACM MobiCom 2021

Zhaowei Tan, Boyan Ding, Zhehui Zhang, Qianru Li, **Yunqi Guo**, Songwu Lu. Device-Centric Detection and Mitigation of Diameter Signaling Attacks against Mobile CoreIEEE CNS 2021

Yuanjie Li, Chunyi Peng, Zhehui Zhang, Zhaowei Tan, Haotian Deng, Jinghao Zhao, Qianru Li, **Yunqi Guo**, Kai Ling, Boyan Ding, Hewu Li, Songwu Lu, Experience: a Five-Year Retrospective of MobileInsight, ACM MobiCom2021

Yunqi Guo, Zhaowei Tan, Songwu Lu. Towards Model-Centric Security for IoT Systems, IEEE ICCCN 2020 (Invited paper)

CHAPTER 1

Introduction

As we initiate this dissertation, we confront an immediate and pressing issue: the communication barrier between the deaf and the hearing. Globally, there are more than 70 million deaf people [Nat23]. Despite the advancements in communication technologies, a holistic solution to effectively bridge this chasm remains challenging. Communication between sign language and vocal language is challenging because of the lack of interpreters and translation tools [Inc21]. Existing approaches are often found lacking in accessibility and operational capabilities. With the advent of AI algorithms that can interpret hand and body gestures, a new era of communication is within reach. Our research centers on the emerging field of Assistive Augmented Reality (AR) systems and their potential to interpret sign language with human-like comprehension.

At the core of our exploration lies the pioneering theory put forth by William Stokoe [STO60], which revolutionized our understanding of sign language. Stokoe's theory asserts that sign languages are legitimate languages with distinct syntax and morphology, as opposed to being merely gesture-based systems. This fundamental principle has served as our guiding light in leveraging linguistic research on American Sign Language (ASL) to advance the field of automatic sign recognition with mobile/AR systems. Through our efforts, we aim to unlock unprecedented possibilities for enhancing accessibility and communication.

This dissertation's central premise is that the divide between the deaf and hearing communities can potentially narrow through well-conceived, systematic solutions. Our first focus is on high-stakes situations such as emergency services. Our research reveals that transla-

tion can be effectively accomplished by classifying signs into specific parameter categories. This understanding, coupled with the recognition that ASL has its own unique syntax, albeit different from English, allowed us to extend our focus beyond emergency scenarios and into more generalized communication settings by converting ASL dictionaries into machine-readable formats.

Recognizing the dynamic and evolving nature of all languages, including ASL, we emphasize the need for continuous learning and interaction with sign language users. Therefore, we are presenting a design to generalize the assistive system for use in a wider range of cases. We are also developing AnySign, which is a platform that provides ASL interaction capabilities, collects signs, offers full sentence translation, and includes interactive features. With AnySign, we hope to move one step closer to achieving truly inclusive communication.

1.1 Challenges in Bridging the Deaf and Hearing Divide

As we delve deeper into the task of connecting the deaf and hearing communities, we encounter two substantial obstacles that require our attention and inventive solutions.

Visual Modality in Sign Language Representation The first major obstacle lies in the scarcity of data. This scarcity is primarily attributed to the absence of a universally acknowledged written form of sign language. The lack of written records has led to an increased reliance on video-based documentation for sign language, a resource that only recently became widely accessible with the advent of online video platforms such as YouTube.

When placed side by side with textual and audio data, the resources available for learning, recognition, and translation systems for sign language are noticeably sparse. This deficit of data is not a mere inconvenience but a formidable roadblock that impacts several dimensions of sign language understanding and machine learning.

Firstly, data scarcity directly constrains our capacity to train robust, high-performing sign recognition models. Without a rich and diverse data set, these models might struggle to

understand and accurately interpret the complexities and nuances of sign language. Secondly, the lack of substantial data sets also limits our ability to provide comprehensive sentence-level translation for sign language. Lastly, this data deficiency makes it challenging to generate correct sign production, particularly given the unique syntax features of sign language.

This lack of data presents a multifaceted problem, making the quest for effective communication between the Deaf and hearing communities more challenging.

Constraints in Sign Language Capture and Representation The second hurdle relates to the technological limitations in capturing and representing sign language in an accessible and user-friendly manner. Given the visual-gestural nature of sign language, which involves hand gestures, facial expressions, and body postures, its effective capture and production necessitate advanced interaction with visual content.

A variety of advanced technologies and specialized equipment, such as sensory gloves [OL11, MK02], RGB-D cameras [JVH19, AGR14], and RF devices [GGM20], have been explored for sign language capture. However, these solutions often face practicality issues. Their limited availability and potential incompatibility with routine scenarios hinder their widespread adoption and diminish their practical impact.

Additionally, rendering sign language requires delivering clear and discernible visual content. Presently, we are faced with a shortage of solutions that can seamlessly present sign language without causing distraction or hindrance to users.

The majority of sign languages, including American Sign Language (ASL), British Sign Language (BSL), and other national or regional sign languages, predominantly rely on in-person communication. This approach is largely due to the visual-spatial characteristics of sign languages, where the three-dimensional space, handshapes, movements, and orientations collectively convey meaning. As a result, the teaching and learning of sign languages have been primarily conducted in person, which allows learners to accurately perceive and practice these visual and spatial aspects.

1.2 Our Contribution

This thesis focuses on the critical issue of communication barriers faced by the Deaf community in our interconnected society. Factors like lack of accessibility, insufficient support for sign languages, and a dearth of efficient translation resources compound these challenges. Our work endeavors to tackle these issues by presenting Assistive AR as a novel solution for bridging the gap between the Deaf and hearing communities. We propose a variety of unique tools and platforms including Sign-to-911 for emergency situations, a general sign language translation attamp, and AnySign, a comprehensive cross-device platform. Utilizing AR as a cutting-edge tool for communication, our goal is to offer solutions that are robust, accessible, and efficient. The primary contributions of this thesis are as follows:

Sign-to-911: A Lifesaver for Sign Language Users in Emergency Situations We underscore the critical need for accessible sign language support, particularly in life-threatening situations where swift and accurate communication is of utmost importance. In addressing the shortcomings of existing systems, we present Sign-to-911, an innovative, lightweight system designed to bridge the ASL-English communication gap during emergencies. By integrating AI/ML models with specific ASL linguistic domain knowledge, Sign-to-911 reduces model complexity while ensuring high translation accuracy and speed. Rigorous testing and evaluations validate its effectiveness and ability to handle high-stakes, rapid communication. Importantly, the system operates on mobile and wearable devices without the need for cloud/edge support, offering accessibility under any circumstance. Sign-to-911 represents a significant stride in emergency communication for the ASL user community, meeting an immediate and crucial need.

AnySign: A Comprehensive Cross-device Platform for Enhanced Accessibility We introduce AnySign, a platform designed to improve ASL accessibility by addressing the unique communication challenges faced by the Deaf community. AnySign integrates the Sign Dictionary, English to ASL Translator, and Teach-Me-Sign modules to form a comprehensive

ASL corpus, offering a robust solution for effective ASL communication. A highlight of the platform is SignChat, a chatbot that merges ASL and AI, enabling an unprecedented mode of ASL human-machine interaction. AnySign, therefore, signifies a significant step towards surmounting the challenges of ASL communication and holds promise for future advancements in this domain.

In addition to the above contributions, mindful of the potential privacy and security concerns inherent in machine learning models deployed on wearable/IoT devices, we also propose a safeguarding solution, MORSE. This solution is designed to ensure privacy protection by leveraging a sampling approach that prevents personal or private models from being exposed to cloud platforms. Notably, this protection is achieved without compromising the functionality and trainability of the models. By doing so, we address a critical aspect of machine learning applications in assistive technology, striking a balance between user privacy and the need for data to drive innovation. This approach underscores our commitment to ethical and responsible technology development, ensuring that our solutions not only provide utility and accessibility but also respect and protect the privacy of the users.

1.3 Organization of the Dissertation

The structure of this dissertation is as follows: Chapter 2 provides a comprehensive background on our research context, with a particular focus on the d/Deaf community, sign language, and existing assistive communication solutions. In Chapter 3, we outline our research objectives and the methodology we employed. Chapter 4 presents Sign-to-911, a solution targeting the first need - emergency communication for the Deaf community. Subsequently, we broaden our approach to encompass general communication scenarios by introducing the design with dictionary-based translation in Chapter 5, and AnySign, a platform implemented for extending sign language accessibility in Chapter 6. In Chapter 7, we detail MORSE, a mechanism for protecting privacy in machine learning models deployed on wearable and IoT

devices. Finally, we conclude the dissertation in Chapter 8 with discussions on potential future works.

CHAPTER 2

Background

This chapter introduces the background of the deaf community and sign language communication solutions. We first delve into the cultural dynamics within the d/Deaf community, emphasizing the pivotal role of sign language and the importance of enhancing sign language accessibility. Then, we examine state-of-the-art technologies to bridge the communication gap between the deaf and hearing, including sign language recognition, production, and assistive AR technologies. Significant challenges remain despite strides made. The chapter concludes by exploring the ongoing issues faced by the d/Deaf community, highlighting the urgent need for innovation in creating effective, user-centric technological solutions.

2.1 d/Deaf Community and Sign Language

2.1.1 Distinguishing 'deaf' and 'Deaf'

'Deaf' with an uppercase 'D' and 'deaf' with a lowercase 'd' signify different contexts within the spectrum of auditory variation. The term 'Deaf' with a capital 'D' embodies a cultural identity, referring to individuals who are part of the Deaf community and culture. These individuals frequently use sign language as their primary means of communication, and they share a communal history and set of experiences. Conversely, 'deaf' with a lowercase 'd' is a medical term used to categorize anyone with severe hearing variation, regardless of their identification with the Deaf community or culture. These individuals may rely on various communication methods, such as spoken language, written language, or sign

language, influenced by their personal experiences and circumstances. By distinguishing between 'Deaf' and 'deaf,' this work aims to respect both cultural identity and clinical perspectives, ensuring a comprehensive and respectful exploration within these communities.

The Deaf community constitutes a cultural and linguistic collective of Deaf individuals, unified by common norms, values, and traditions, predominantly centered on the use of sign language for communication [PH88, Lan05]. This cultural identity transcends the audiological status indicated by the lowercase 'd' in 'deaf' [SD74]. Far from being a mere extension of hearing culture, Deaf culture constitutes a distinct entity characterized by a rich heritage, art, storytelling, and humor, profoundly shaped by common experiences and a shared visual language [Lad03].

Building on this cultural foundation, the Deaf community is closely interwoven through robust social connections. These connections are not born out of shared disability but rather shared language and cultural background [MNS03]. A multitude of institutions and events serve as social hubs, including Deaf clubs, schools for the Deaf, and Deaf sporting events. These gatherings offer a space where sign language is the primary method of communication, reinforcing cultural identity and a sense of community.

Despite this shared sense of identity, the Deaf community is marked by significant linguistic diversity [Luc01]. It's crucial to note that there isn't a single universal sign language. Instead, various sign languages exist around the globe, each with its unique grammar, syntax, and lexicon, reflective of the cultural nuances of the regions they originate from [SM02].

2.1.2 Sign Language and Its Roles

Sign language is not a simplified, gestural version of a spoken language but rather a full, complex, and natural language in its own right. Like any spoken language, it has its own grammar, syntax, and lexicon [STO60].

One prevalent sign language used globally is American Sign Language (ASL). ASL, like

other sign languages, comprises five primary parameters: handshape, movement, location, palm orientation, and non-manual signals [Bat78]. Each parameter adds a layer of meaning to the signs and the overall linguistic structure, contributing to the richness and complexity of the language.

Furthermore, the grammatical structure of ASL differs significantly from English. While English primarily adheres to a subject-verb-object (SVO) sentence structure, ASL commonly follows a topic-comment structure. Consider, for example, the English sentence "I like ice cream." In ASL, this would typically be signed as "ICE-CREAM I LIKE," with the topic (ice cream) presented first, followed by the comment (I like) [NKM00].

The role of sign language extends far beyond simple communication; it is a critical component of deaf education, community interaction, and identity formation. In education, sign language offers a more intuitive and engaging way for deaf children to learn, leading to improved academic outcomes [May07]. Moreover, sign language serves as a cornerstone for community interactions within the d/Deaf communities, facilitating social connection and mutual understanding [Pad16]. Lastly, sign language plays a key role in Deaf identity formation, with many deaf individuals viewing sign language use as a source of cultural pride and identity.

2.1.3 The Imperative of Sign Language Accessibility

The significance of sign language cannot be overstated, particularly when it comes to education and employment for deaf individuals. Deaf education has seen an extensive debate between oralism (focusing on spoken language) and manualism (focusing on sign language), with current trends favoring bilingual-bicultural education where both sign language and spoken language are taught. Moreover, mainstreaming, or the practice of placing Deaf students in regular classrooms, further underscores the need for sign language accessibility, as the majority of these classrooms are often structured around spoken language [Che21].

In the realm of employment, deaf individuals continue to face significant challenges, including lower employment rates and limited job opportunities compared to their hearing counterparts [Pun16]. Sign language proficiency often plays a vital role in the career prospects of deaf individuals, enabling them to communicate effectively in the workplace and access equal opportunities.

The importance of sign language accessibility extends beyond the confines of education and employment; it is a fundamental human right. Legislation such as the Americans with Disabilities Act in the U.S. and the UN Convention on the Rights of Persons with Disabilities underscores the rights of deaf individuals to access communication in their preferred language [Dep20]. Such legislation, in theory, guarantees accessibility, including communication access (e.g., interpreters, captioning) and physical access (e.g., alerting devices), thereby facilitating the inclusion of deaf individuals in all aspects of life.

Despite these legal protections, however, significant gaps persist in policy enforcement and accessibility, particularly regarding access to healthcare and emergency services. These ongoing issues underline the dire need for more robust, efficient, and accessible sign language resources and strengthen the argument for technology such as Assistive AR, which has the potential to bridge these gaps and foster greater inclusion.

2.2 Breaking the Communication Barrier: State of the Arts

This section explores recent developments in sign language recognition and translation, production, and assistive AR technologies, all critical components in bridging the communication gap between the deaf and hearing communities.

Sign Language Recognition and Translation Sign language recognition has witnessed a surge of research interest over recent years. Existing solutions generally fall into two categories: vision-based and sensor-based approaches. Vision-based approaches like I3D [LRY20],

SAM-SLR [JSW21], and DeepASL [FCZ17] use cameras to capture and analyze signing motion, while sensor-based methods exploit various devices such as gloves [AZZ18], smart watches [HLZ19], earphones [JGZ21], and EMG sensors [ZJW22]. Both approaches typically employ deep-learning-based models, such as Recurrent Neural Networks (RNNs) and their variants, for sign recognition.

However, these existing solutions share several limitations. Primarily, the complexity of deep-learning models restricts the size of the vocabulary set they can recognize—often around 100 signs. Moreover, they have difficulty with sentence-level translations for American Sign Language (ASL). Some offer no sentence-level translation [LRY20, JSW21], while others rely on complex temporal model training on extensive sentence corpora [FCZ17, AZZ18, HLZ19, JGZ21, ZJW22].

Sign Language Production When it comes to sign language production, several applications have been developed, such as HandTalk [han23] and Sign Language Translator [Sig23a]. Despite their efforts, these apps often struggle to produce sign language with the correct grammar order. They also tend to cover a limited number of signs and depend on clouds/edges for sign production [Viv23, ASL23]. Some even require substantial GPU processing power for rendering and generation [SCH20, SHB20, SCB20].

Assistive AR The advent and advancement of AR and VR technologies in recent years have catalyzed transformative changes across various sectors, with one of the most promising areas being assistive technologies for individuals with disabilities [Dic21]. Innovations in this space have yielded solutions such as navigational aids for people with visual impairments [ZBB18, BH20], and tools that employ AR for enhanced social and emotional learning [VOI23]. The capacity of AR to provide an immersive, interactive, and customizable user experience makes it uniquely suited for applications that aim to bridge gaps in accessibility.

Despite the significant strides taken in the AR field, there remains untapped potential in leveraging these technologies to support sign language communication. As AR technologies continue to mature and become more integrated into daily life, they present a promising avenue to address the communication barriers faced by the d/Deaf community. The integration of AR with sign language communication not only represents a novel application of this burgeoning technology, but also has the potential to revolutionize the way the d/Deaf and hearing communities interact, thus creating more inclusive environments.

In summary, while progress has been made in various domains related to sign language recognition, translation, production, and assistive technologies, there remain substantial challenges to be addressed in enhancing accessibility and facilitating effective communication between the d/Deaf and hearing communities.

2.3 Challenges in Sign Language Accessibility and Technology

Despite advancements in technology, the d/Deaf community continues to face prevalent communication challenges. A crucial issue is the general public's lack of understanding and awareness of sign language. This gap in knowledge can lead to exclusion and discrimination, limiting opportunities for deaf individuals to interact and engage effectively with the wider society. Additionally, access to sign language interpreters remains inadequate. The limited availability and high cost of professional interpreting services often pose substantial barriers, making routine tasks such as medical appointments, educational events, or even casual social interactions challenging for deaf individuals. Barriers also persist in accessing digital content, given that much of the online content lacks proper sign language interpretation or closed captioning, leading to an information gap.

While technology has the potential to address these challenges, it has yet to fully bridge this divide. The development of reliable, easy-to-use tools for sign language translation and interpretation is still a work in progress, with many current solutions falling short in terms

of accuracy, comprehensiveness, and usability. As such, there is an urgent need for continued research and innovation in this domain, with a particular emphasis on developing technologies that are not only functional but also align with the specific needs and preferences of the Deaf community. Advances in areas such as machine learning, natural language processing, and augmented reality provide promising avenues for this work, with the potential to significantly enhance communication accessibility and inclusivity for deaf people.

CHAPTER 3

Overview

The communication divide between Deaf and hearing individuals presents a compelling challenge in computer science, underscored by the acute lack of widely accessible and effective solutions. This research is dedicated to filling this gap, concentrating on the development of practical, accessible technologies specifically crafted to dismantle the communication barrier. Leveraging inclusivity as a cornerstone, this endeavor strives to significantly contribute to a more connected society by fostering seamless interactions between Deaf and hearing communities.

3.1 Research Objectives

The core objectives of this research are anchored in three distinct, yet interconnected, technical advancements aimed at enhancing Deaf-hearing communication:

Sign-to-911: This initiative addresses a critical issue—the need for swift, accurate sign-oral language communication during emergencies. The objective is to create an accessible, real-time translation mechanism that enables effective sign language communication during high-stake situations.

AnySign: This component of the research extends beyond emergency scenarios to the broader context of daily communication. AnySign's goal is to build an open, inclusive platform that facilitates learning, interaction, and accessibility in sign language. Its design promotes a more general application, offering users the ability to engage more fully in various

social and professional contexts.

MORSE: Acknowledging the increasing prevalence and utility of wearable and IoT devices in our society, MORSE aims to deliver a solution for model privacy protection. The objective is to safeguard user privacy by mitigating the risk of exposing personal data when using cloud services, while preserving the functionality and trainability of the models.

Each of these objectives addresses a unique aspect of the communication barrier. Together, they form a holistic approach to bridging the Deaf-hearing communication gap.

3.2 Methodology: Towards Practical and Accessible Solutions

Our research methodology involves a domain-oriented approach that underscores sign language recognition, translation, and production. During our study, we have fostered collaborations with the Deaf community and linguistic experts specializing in ASL. Such interactions have been instrumental in gaining valuable insights into the challenges faced by the community and the practical solutions that can be widely accepted by potential users.

Our focus has been on delivering accessible solutions that seamlessly integrate into everyday life. With this objective, we have chosen assistive AR glasses as our primary hardware setting, an everyday wearable that aligns with our vision of accessibility. The chosen hardware has several advantages:

1. The glasses function as a standalone device, reducing the need to connect to other devices such as smartphones, thus facilitating ease of use.

2. Equipped with sensors and a display, the glasses provide a means to capture signs and project translated information, thereby serving as an efficient tool for bridging the communication gap.

Our research remains committed to achieving bidirectional communication, acknowledging the fact that the communication barrier is a two-way issue. Consequently, all our pro-

posed techniques aim to not only translate sign language for the hearing, but also to convey oral language information to Deaf users, thereby realizing a holistic communication environment.

3.3 Roadmap

The remainder of this dissertation is structured as follows: Chapter 4 introduces Sign-to-911, a solution developed for enhancing emergency communication between the Deaf and hearing communities. We discuss its design, implementation, and effectiveness in this chapter. Chapter 5 and Chapter 6 extends our efforts to general communication scenarios, where we introduce AnySign, a comprehensive platform aiming to democratize sign language usage and increase its accessibility. Lastly, Chapter 7 presents MORSE, a mechanism we designed for privacy protection in machine learning models used on wearable devices and IoT systems. This chapter elaborates on the technical details of this private model protection method.

CHAPTER 4

Sign-to-911: Emergency Call Service for Sign Language Users with Assistive AR

In the United States and parts of Canada, American Sign Language (ASL) is the primary means of communication for individuals with hearing disabilities, encompassing an estimated 500,000 to 2 million users [MYB06]. ASL is unique; as a visual language, it depends on the intricate movements and formations of the hands to express thoughts and ideas. This distinctive aspect of ASL presents both challenges and opportunities, particularly when it comes to translating ASL into English or other languages, such as in critical emergency situations like 911 calls. Through our survey of 54 ASL users, we discovered that the current options for 911 services—either typing messages to a 911 operator or using a video relay service—are deemed inconvenient and not readily accessible.

In this chapter, we set out to address an essential question within the ASL community: Can we construct a compact mobile system solution capable of translating ASL into English, operating on wearable devices and smartphones, without the need for cloud or edge support? If possible, this would facilitate real-time, direct emergency call service between an ASL user and a 911 operator, each using their preferred languages. Although recent advancements in AI, vision, and machine learning may be beneficial, they must be adapted to operate on mobile and wearable devices without causing excessive delay or overprocessing. Unfortunately, this restriction eliminates most deep learning-based proposals [LRY20, JSW21], as they cannot run on mobile devices without incurring excessive delay and processing.

We present a solution that deviates from the typical deep learning paradigm and focuses

on designing a lightweight system, Sign-to-911, which facilitates swift ASL to English translation. This system utilizes a pair of assistive Augmented Reality (AR) glasses in tandem with a smartphone. The AR glasses capture live videos of the user's sign motions during an emergency 911 call, forwarding the video frames to the smartphone via Bluetooth for sign recognition and sentence translation. The translated English texts are then vocalized and relayed to the 911 operator, while voice responses from the operator are converted back into ASL sentences, displayed as sign animations by a 3D avatar on the AR glasses.

Sign-to-911 capitalizes on traditional AI/ML models, offering simplicity and fewer model parameters, significantly reducing the complexity compared to recent deep learning models for ASL sign recognition. By treating ASL signs not as random hand gestures, but rather structured motion patterns following the syntax rules of ASL, we significantly simplify the design. Leveraging recent algorithms from graphics, vision, natural language processing (NLP), and AI/ML, we strive to achieve fast and accurate recognition and rendering at both the sign level and sentence granularity.

This chapter details the implementation of Sign-to-911 on standard AR glasses and Android phones, presenting an affordable, accessible solution that recognizes a broader spectrum of distinct signs and supports fingerspelling, a necessity in emergency situations. Our evaluation with six ASL signers demonstrates encouraging results, with our models achieving impressive accuracy rates and an average end-to-end latency of 0.55 seconds for 550 signs, a significant improvement over running prior proposals directly on smartphones [JSW21]. This chapter further elaborates on the details of the Sign-to-911 system, signifying a pivotal step toward bridging the communication gap for the ASL community.

This chapter presents our expedient and efficient solution, Sign-to-911, aimed at facilitating immediate ASL-English communication in emergency situations using assistive AR. In §4.1, we explore the challenges present in emergency communication for the deaf and hard-of-hearing communities. We present an overview of our solution in §4.2. The specifics of Sign-to-911's innovative design are thoroughly discussed in §4.3.1 and §4.4. We delve into

the implementation process in §4.5 and assess its effectiveness in §4.6 examines the potential pathways for deploying Sign-to-911 within existing emergency communication frameworks. In §4.7, we discuss the signing variants and the compatibility of our solution on different hardware and use cases. We review relevant works in the field in §4.8 and summarize the chapter in §4.9.

4.1 Emergency Calls for Deaf People

911 provides emergency call service in the US. While offering multi-language support (e.g., English/Spanish, etc.), the current 911 system does not supply an efficient communication channel for ASL users to communicate with 911 operators. This is largely due to the gap between ASL and other spoken languages; ASL is a visual communication language that requires runtime viewing for correct interpretation.

To bridge the gap, sign language users conventionally use two alternative schemes to make their emergency calls: text-based communication or relay services. Text-based communication services, such as real-time-text (RTT) and teletypewriter (TTY), transcribe voice to text and allow ASL users to input text during a call. Text-to-911 allows individuals to send text messages to emergency services. However, these services are not applicable to many deaf individuals since not every sign language user has the same level of proficiency in a written language (say, English).

The other approach is to use relay services, such as video relay service (VRS), during a call. The user streams video to an interpreter who translates the sign language or into a voice message for the 911 operator. However, this approach requires high-speed network for video calls, which may be unavailable during emergencies. Moreover, the interpreter shortage makes this approach difficult to scale [The22].

User Survey. We conduct a survey to learn about the 911 call experience for ASL users. We collect responses from 54 volunteers in the ASL community via anonymous online ASL

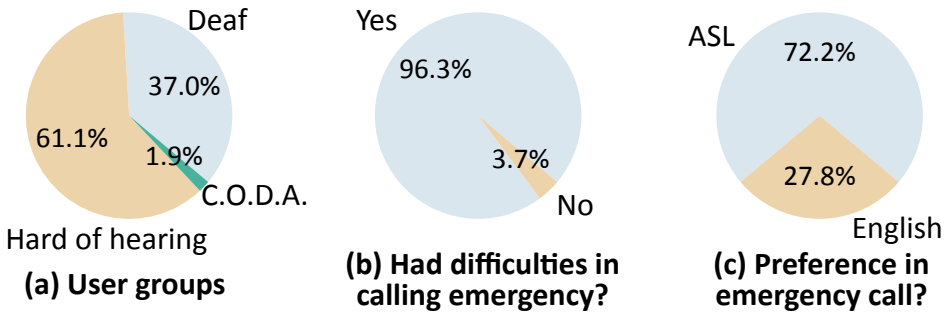

Figure 4.1: Emergency call survey for ASL users

forums. Participation are open to all community members without any material/financial incentive. As shown in Figure 4.1, among the participants, 37.0% are deaf, 61.1% are hard of hearing, and 1.8% are children of deaf adults (CODA). The survey shows that 96.3% of them have experienced difficulties communicating with an emergency call operator, mainly because TTY or VRS services are not readily accessible. Individuals who can speak but are deaf or hard of hearing often have to repeat their addresses and situations until the requested help arrives due to a lack of feedback. On the other hand, those who cannot speak are unable to make 911 voice calls. Furthermore, 72.2% of the participants prefer to communicate in ASL during emergency calls, since ASL is their primary language in daily life.

Goals. The survey result motivates us to devise an effective solution for the ASL community to make emergency calls. Specifically, the solution should have the following features: (1) Accurate bi-directional translation: it must support accurate, two-way communications between the signer and a 911 operator: ASL signs-to-spoken English and spoken English to ASL signs. (2) Fast translation: the bi-directional translation must be fast enough to ensure liveness and interactiveness of the call conversation [MZB21]. (3) Ease to use and carry: the solution should be easy to use and carry with the signers, since a significant portion of emergency situations arise on the road, or at remote or not readily-accessible locations. (4) Operation in the absence of high-speed Internet access: we do not assume infrastructure support for real-time video transfers and interpretation of ASL, except for a conventional voice call service. This is a common scenario for emergencies in remote or not readily accessible

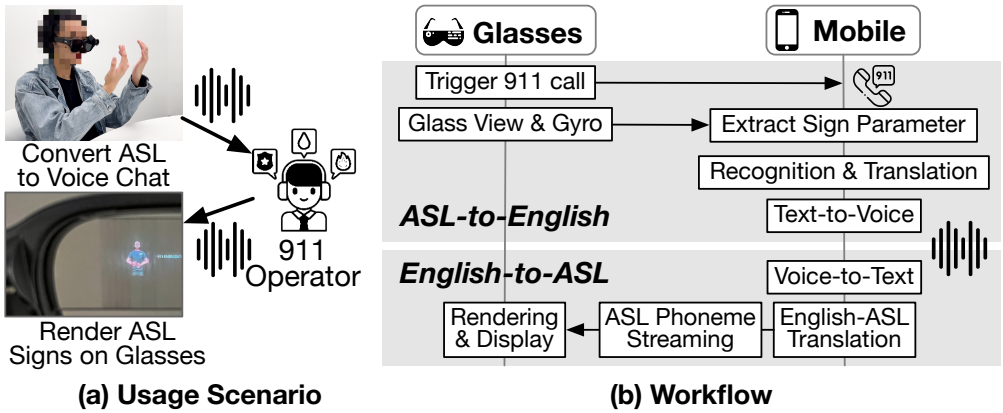

Figure 4.2: Workflow of Sign-to-911

regions.

Limitation of Current Machine Translations for ASL Signers. The current machine-based solutions for ASL signers, either software-centric or hardware-based, cannot meet all goals and well serve the emergency call scenarios. The fundamental problem is that, ASL is a visual language for communication in nature, and any translation scheme must capture and recognize each sign (or fingerspelling) in an accurate and timely fashion.

Existing software solutions, such as SL-GCN [JSW21] and I3D [LRY20], heavily rely on computer vision techniques. Therefore, they require high-end GPUs for fast processing, thus unsuitable for signers without access to cloud/edge services. Network communications with cloud/edge servers may incur long latency and compromise call interactiveness.

The proposed hardware solutions, such as gloves [ZCL20] and smartwatches [HLZ19], capture and recognize signs with sensors and hardware processing. However, they are deemed impractical for everyday wear and lack the necessary resolution to fully capture sign features. Moreover, they only provide uni-directional communication, which does not fit the two-way communication between the caller and the 911 operator.

4.2 System Overview

We now describe Sign-to-911, a lightweight solution that possesses all four features and provides emergency call services between ASL signers and 911 operators.

The system components and workflow for Sign-to-911 are shown in Figure 4.2. In Sign-to-911, the signer wears an assistive pair of Augmented Reality (AR) glasses, which interact with his/her smartphone. With a click on the 911 icon on the glasses, the user makes an emergency call, which is initiated through his/her smartphone. The smartphone subsequently works with the AR glasses for sign-to-English translation. The translated spoken English will be sent to the 911 operator via the established 911 call. The voice response from the operator will be translated to ASL signs, which are further rendered on the AR glasses. It thus offers bidirectional call communication between the ASL signer and a 911 operator.

We select the AR glasses (illustrated in Figure 4.2), rather than other wearable hardware (e.g., gloves or smartwatches), for ASL signers. They have a lightweight design and non-distractive displays, and limited on-glass processing capability, thus allowing users to wear them like sunglasses or normal eyeglasses. They typically have a built-in camera, a color display, a gyroscope sensor, and a speaker (for music listening), as well as a Bluetooth interface that readily connects to nearby smartphones at speed up to 150KB/s [Tra23]. The assistive glasses keep a signer's hands free for signing, while allowing him/her to see the operator's responses in ASL sign animations on the display. Moreover, assistive AR glasses of this type are quite affordable, with current prices ranging from 350 USD [INM23] to 1300 USD [Vuz23].

Our wearable system takes a software-centric approach. We thus design two pipelines to enable two-way communications: ASL-to-English and English-to-ASL (Figure 4.2). On the ASL-to-English translation, we capture each ASL sign by exploiting the sign parameters and simple machine-learning models. These parameters also provide a standard description of the signs. We thus enable fast recognition of signs based on the ASL domain structures. Once

signs are recognized, we leverage the emergency call context and syntax model to convert sign sequences to English sentences. On the English-to-ASL translation, we first translate each English sentence to the corresponding ASL gloss with the syntax model. We then produce the ASL sentence using the basic language unit: phoneme. We further compress phonemes leveraging kinematic correlations, in order to feed the ASL streaming into the low-rate Bluetooth connectivity. Finally, the glasses decompress the phoneme stream and render the ASL signs and sentences in front of the user's eyes.

Given the glass-smartphone setup and simple workflow for each user, we address two key challenges with novel domain-driven designs to enable ASL-to-English translation and English-to-ASL production.

Accurate and Fast Sign Translation. A primary challenge in our Sign-to-911 scheme is to capture and recognize each ASL sign without relying on edge/cloud infrastructures. Sign language is a natural, visual language that is conveyed through a sequence of gestures. We thus have to use machine learning models that can analyze a substantial number of sign features in a timely fashion. This may in turn increase model complexity further. Note that the used models must be processed on mobile and wearable devices in our system setting. Consequently, it remains difficult to develop and operate a lightweight solution that translates sign language with high fidelity at runtime. To address this issue, we depart from the popular paradigm of deep-learning based models. Instead, we propose a novel method that exploits sign parameters derived from linguistics and extensive domain knowledge in ASL. Our method enables us to capture sign gestures efficiently and ensure accurate and fast recognition.

Efficient Sign Production from English. Many ASL users consider ASL to be their primary language. However, producing ASL signs from English sentences and rendering them on AR glasses can be challenging. Each sign involves the coordinated actions of fingers, palms, and arms. Moreover, the AR glasses do not have enough processing capability, and the Bluetooth connectivity between the glasses and the smartphone cannot transfer and

render each produced sign gesture on time. Our proposed solution utilizes the phonetic parameters of ASL to generate accurate signs. We further leverage kinematic correlations for efficient sign compression. Consequently, we achieve high-fidelity ASL production on smartphones and accurate rendering on glasses.

4.3 ASL-to-English Translation

Our first task is to provide swift and accurate translation of ASL into English during a 911 call given the glass-mobile setting. Figure 4.3 shows the three main steps. First, we capture the sign parameters for recognition (§4.3.1). Second, we leverage ASL domain knowledge to perform fast sign recognition with the parameters (§4.3.2). We further construct coherent sentences from the sign sequences and translate them into English (§4.3.3). Our lightweight translation pipeline works with limited capabilities of smartphones and AR glasses.

Our approach differs from all prior proposals [FCZ17, LRY20, JSW21]. To recognize signs from visual information, the conventional approach treats signs as a spatial-temporal sequence of hand movements and uses computer vision techniques for sign recognition. However, these methods often result in large, complex models that require significant computational resources for both feature extraction and recognition. For example, I3D [LRY20] uses video as its input and requires 4.8GB GPU memory. SL-GCN [JSW21] takes skeleton sequences as input but still requires 16 million parameters and 5 seconds to recognize one sign with a powerful GPU. Consequently, these models are not suitable for this setting.

4.3.1 Capturing Sign Parameters

We devise a novel scheme to effectively extract sign parameters from the video frames capturing the signer's gestures by the AR glasses. Our approach is based on the premise that ASL is a visual, yet natural language with linguistic features similar to those found in any spoken language. Like the consonants and vowels used in spoken languages, ASL signs follow

a set of decomposable patterns and rules (i.e., sign factors). Such factors set the foundation for differentiating signs from arbitrary human gestures. By identifying and analyzing these patterns, we can effectively capture and interpret ASL, conceptually similar to how a spoken language is transcribed and analyzed.

Linguistic factors for ASL signs In general, ASL signs are classified into one-handed and two-handed signs. It further uses four main linguistic factors to define a sign: handshape, palm direction, location, and hand movement [TGB10][1]. The handshape describes the configuration a hand assumes when making a sign. The ASL dictionary lists 40 handshapes to organize signs [TGB10] (see Figure 4.4 for illustrative examples). Palm orientation indicates the orientation on how the palm of a hand is turned. It has six choices (up, down, left, right, front, or back) in ASL. The location specifies where a sign is formed. It is expressed with respect to close body areas when a sign is performed (say, "in front of face," "near right ear," "right cheek," etc.). Movement specifies the direction and trajectory of how a sign moves in space. It also includes repetition, motion magnitude, and speed.

Note that hand movements are not arbitrary in ASL, either. They exhibit three patterns [Cou14]: (1) based on the movement directions, signs can be classified as static (a.k.a. fingerspell), unidirectional and repeated signs; (2) for two-handed signs, both hands may move or only the dominant hand moves; (3) if both hands move, the movement pattern can be classified as symmetric, parallel, or alternating (Figure 4.5).

In summary, from the ASL linguistic standpoint, each sign can be well specified using the above four main factors. Furthermore, each linguistic factor only assumes a limited number of choices for defining a sign.

Issues with direct utilization of ASL linguistic parameters However, we cannot directly use the above four sign factors to capture and recognize a sign. There is a nontrivial gap between the factors used by linguistic studies and those sign parameters that ML-based

[1]Nonessential parameters, such as body and facial expressions, provide additional information such as tone.

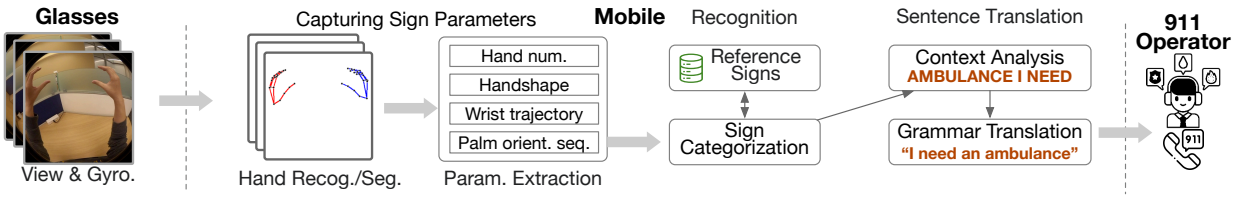

Figure 4.3: ASL-to-English Pipeline

schemes may use to capture and recognize signs.

First, the 40 handshapes are well defined by the ASL dictionary [TGB10]. However, most proposed systems cannot obtain such handshape information directly. Instead, they only have access to the visual information obtained through the sequence of images or video frames when a signer makes a sign (say, through the AR glasses in our case). Second, location (with respect to close body parts, e.g., "below right shoulder") is an important linguistic factor to define a sign in ASL. This is also deemed to be difficult in reality, since sensors (such as AR glasses) cannot capture the full view of the signer's body. Therefore, they cannot learn the relative positioning of body parts versus hand sign. In fact, we are unaware of any prior scheme that uses the location factor in ASL to classify a sign.

We thus have to design a set of sign parameters for our ML schemes, while leveraging the ASL linguistic factors. Such parameters must be readily obtained on mobile devices without intensive processing, and enable fast parameter extraction and sign recognition.

Sign Parameters for ML. In order to facilitate accurate recognition and translation of ASL using machine learning algorithms, we have identified four essential types of sign parameters. These parameters are specifically designed to be independent of the signer's body shape and hand size, ensuring robustness and consistency across different signers. By incorporating these sign parameters into our ML models, we aim to enhance the accuracy and reliability of ASL-English translation. Table 4.1 presents a comprehensive overview of the four sign parameters that are crucial for our ML algorithms. Each of these parameters captures unique aspects of ASL signs, enabling our system to accurately recognize and interpret the intended meaning of signs. Let us now delve into the details of each of these

Figure 4.4: Basic handshapes from glass view (40 in total)

parameters:

1. **Hand number:** This parameter is represented as a two-dimensional vector that denotes the probability of a sign being performed with one hand or two hands. By analyzing the hand configuration, our ML algorithms can discern whether a sign is executed using a single hand or both hands

2. **Handshape:** The handshape parameter captures the sequential variation of hand configurations over time. At each time step, the handshape is represented by probabilities associated with 40 basic ASL handshapes (Figure4.4).

3. **Wrist trajectory:** Capturing the trajectory of the wrist is essential for accurate sign recognition and translation. The wrist's coordinates (x, y, z) are tracked over time t, utilizing hand detection and depth estimation techniques that account for the signer's hand size. This parameter provides valuable information about the movement and positioning of the wrist.

4. **Palm orientation sequence:** The palm's orientation is a vital aspect of ASL signs, as it conveys important linguistic and grammatical information. The palm orientation sequence parameter captures the variation of the palm's orientation (α, β, γ) over time t.

These four parameters provide an ML-friendly representation of signs. We next explain how to extract these parameters on the glass-mobile setting.

Parameter	Meaning	Dimension #
Hand number	One-handed or two-handed	2
Handshape	Handshape sequence over time	$40 \times 2 \times t$
Wrist trajectory	Wrist position over time	$3 \times 2 \times t$
Palm orientation seq.	Palm direction over time	$3 \times 2 \times t$

Table 4.1: Sign Parameters

Sign Parameter Extraction. We extract the above sign parameters from the video frames captured by the AR glasses. The extraction procedure has three steps: skeleton extraction, segmentation, and adaptive extraction.

Skeleton Extraction. Extracting 3D hand skeletons from video frames is commonly supported by mobile libraries [Med23] on smartphones. However, on-glasses processing can only achieve 3 FPS, much lower than the natural signing speed. We thus offload processing to the signer's smartphone to extract hand skeletons. Since the signer wears the AR glasses as (s)he makes a sign, one new issue arises: head movements affect skeleton positioning in the camera view. We leverage the gyroscope data to address this issue. With such data being further streamed from the glasses to the phone, the signer's hands can be calibrated relative to a fixed position, thus reducing the impact of head movements.

Segmentation. The captured wrist trajectory sequences need to be segmented into signs. We exploit the idle states of the hands in segmentation. Prior ASL study [Cou14] shows that, pause and neutral position are critical to identifying pacing between sequential signs. Specifically, the signer may pause after the current sign, before transitioning to the next sign. Alternatively, the signer's hands may return to a neutral position (i.e., where the hands remain relaxed, typically at the waist level in front of the body [TGB10]), before starting the next. We thus use the pause time and the hand's neutral position to detect the borderline of sequential signs. The sign parameters are then ready for final extraction.

Adaptive Extraction. The hand number and movement can be directly extracted from the hand skeletons. Palm orientation can be computed from the normal vector of the palm plane. The extraction of handshape from each frame is more involved. We first calculate

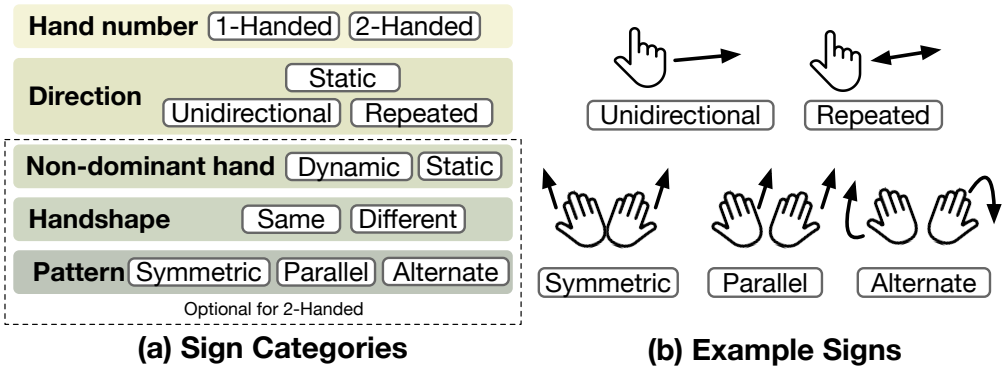

Figure 4.5: Sign Categories and Examples

the angles of finger joints [ZBV20] to minimize the impact of different hand sizes among signers. We then match joint rotations with the 40 base ASL handshapes using a multilayer perceptron classifier[HTF09], which assumes a hidden layer of size 64. Since each hand may contain one or two handshapes during the sign [Fri76], we further merge the handshape sequence to four vectors (two for each hand-start and end). This merging process helps to reduce recognition errors that may occur with individual frames.

To further optimize processing at the smartphone, we design an adaptive handshape extraction scheme. Our approach involves taking different sample rates on dominant and non-dominant hands for handshape recognition. In ASL, the dominant and non-dominant hands have different impacts on sign meaning. The non-dominant hand typically undergoes fewer changes in position and shape than the dominant one, thus requiring lower sample rate. A signer may set his/her dominant hand via configurations; the default dominant hand is the right hand. This adaptive extraction further reduces processing time by about 25%.

4.3.2 Sign Recognition from Parameters

The signer's gestures are thus converted into a sequence of sign parameters. The next task is to perform sign recognition from such parameters. Note that our recognition must classify word signs from fingerspelling-based terms.

We take a two-step approach. We first categorize a sign candidate into one of a few cate-

gories. We then compare the similarity of the collected sign parameters with all signs in the category. This two-level, hierarchical recognition scheme scales better than flat recognition, where parameters of a new sign are directly compared with all candidate signs (i.e., hundreds or thousands of signs). It reduces processing complexity and enables fast recognition on mobile devices.

Sign Categorization. As shown in Figure 4.5(a), signs are first classified along five dimensions: (1) *Hand number:* signs are first classified into 1-handed and 2-handed with the hand number parameter; (2) *Direction:* using wrist trajectory, signs are classified into unidirectional, repeated, and static. Note that, the static sign (a.k.a. fingerspell) denotes a single letter in ASL; a corresponding fingerspelling module (to be elaborated next) is triggered for further recognition; (3) *Non-dominant hand behavior:* we further decide whether the non-dominant hand is dynamic or static, by using the wrist trajectory of this hand; (4) *Handshape*: the handshapes of both hands are further identified based on the 40 candidate handshape set; (5) *Patterns:* signs are finally classified by movement patterns mandated by ASL: symmetric, parallel, and alternate. The movement patterns are extracted by combing the wrist trajectory and palm orientation sequences. Figure 4.5(b) shows a few examples of different sign categories. The above categorization can be readily obtained from the sign parameters during the training phase, and stored in a reference sign database.

Our evaluation shows that, the above step could reduce the search space by an order of magnitude. The sign category enables us to recognize signs accurately, even when there are slight variations in sign parameters due to differences in signing habits and speeds.

Recognition of Word Signs. We next use a fast dynamic time warping algorithm [SC07] to match the captured hand trajectory with those candidate signs in the same category, and sort out the Top-k candidates based on the weighted similarity of all sign parameters. The weights are learned with a linear regression algorithm to suppress noises. Our evaluation reveals a promising outcome: when k is set to 5, 96.5% of signs are accurately identified, demonstrating a high degree of top-5 accuracy. The accuracy can be further improved using

context information (§4.3.3). This inclusion of contextual factors allows the system to better understand the overall meaning of the sentence, leading to a more precise sign selection.

Recognition of Fingerspelling-based Terms. We exploit two distinctive features of fingerspelling to classify such terms from word signs in ASL. Fingerspelling uses a single hand and does not move the wrist position. Moreover, the whole word, but not individual letters, must be fully expressed with fingerspelling [SDK18].

Our recognition module for fingerspelling thus works in two steps. Each letter is recognized from each frame using palm orientation and handshape. Multiple letters are then merged into a word. Since ASL fingerspelling uses wrist movement to indicate repeated letters in the word (e.g., Z-O-O), we apply wrist movement checker to decide the repeated letters. We further call the English spell checker [Son23] to minimize misrecognition. Our system is designed to trigger fingerspelling recognition under two situations: (1) during a conversation, such as when a user needs to sign a name, and (2) when the system fails to recognize a sign. In the first case, fingerspelling recognition is initiated by placing the dominant hand in a fixed position in front of the glasses. For the second case, the system utilizes confidence scores to detect uncertain recognition and allows the user to correct signs using fingerspelling.

4.3.3 Sentence Translation

Given the recognized ASL signs, we next translate them into an English sentence. We address two issues in the translation. First, ASL follows its own grammar. The grammar differences arise in two main aspects: different word order and simplified structure. For the basic English word order of subject-verb-object (SVO), it often shows up as subject-object-verb (OSV) in ASL. For example, the sentence "I need an ambulance" in English SVO order is translated into "AMBULANCE I NEED" in ASL. Moreover, ASL adopts a simplified structure. For instance, ASL does not have "be verbs" (i.e., am/is/are). ASL also does not use separate signs for articles (a/an/the). To solve the grammar issue, we have devised a

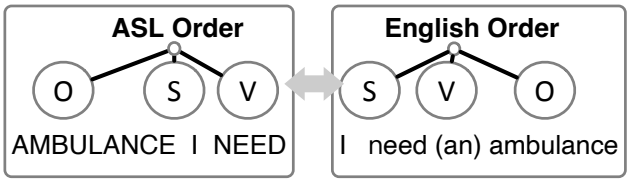

Figure 4.6: Grammar Mapping

English: I donate blood every three months.
ASL: EVERY THREE MONTHS BLOOD I DONATE.
English: The bank near my house was robbed two times.
ASL: TWICE BANK NEAR MY HOUSE ROB.
English: My house has a good security system.
ASL: GOOD SECURITY SYSTEM MY HOUSE HAS.

Figure 4.7: English-ASL parallel corpus

grammar translation model that converts ASL sign sequences into an English sentence. The model works even though signers may have different signing habits in terms of word order. The second issue is that ASL signs may appear as homophones, which assume identical sign parameters but convey different meanings [Hom23]. To address this issue, we disambiguate such signs using context information on the usage scenario. A context model is used to identify the correct sign out of multiple choices in the given context.

Grammar Translation. For grammar translation, we build an ASL syntax model to provide mappings between the ASL grammar orders and the corresponding English grammar orders, such as the "OSV↔SVO." To train the model, we extract the ASL syntax order and map it to the English syntax order using a parallel corpus, which includes the ASL-English translations. During grammar translation, our model first parses the sign sequence and identifies the best alignment of its syntax order. For example, as shown in Figure 4.6, the incoming signs "AMBULANCE I NEED" are mapped to the order "OSV." The model then maps it to the English grammar order "SVO." Second, it fills missing elements in ASL, such as the "be verbs" and articles. If no exact match is found in the known ASL order, the most similar order matching the incoming sign sequence will be applied. This can be done by matching the subtree structures [BKL09].

Our syntax model is trained using a public parallel corpus from the authoritative ASL resource Signing Savvy [Sig23b]. The dataset contains 1233 translations between ASL gloss sequences and their corresponding English sentences. Figure 4.7 shows some example sentences. We thus have learned 209 mappings between ASL and English. This process results in an accurate and efficient translation that produces grammatically correct English sentences from ASL sign sequences.

Leveraging Context Information. Our system further leverages 911 context information from the real-life emergency call conversations to refine the translation, as well as recognition of uncertain signs. We construct our context model using the following process: (1) classify the 911 conversations into five emergency types based on the answers to the type of emergency, (2) track questions asked by the 911 operator, and (3) create a context model using the tuple of [Topic, Question, Response]. The weights of sign candidates are then adjusted based on each tuple value. For example, if the question is on the color of the victim's clothes, the context model selects color-related words in the recognized candidates as the final response. By incorporating contextual information into our recognition and translation, we improve the system accuracy. This becomes important in cases where signs are ambiguous or unclear.

After the translation, the resulting English sentence is streamed to the glasses via Bluetooth. The AR glasses generate corresponding voice message, which is fed in by the ongoing call at the smartphone and sent to the 911 operator.

4.3.4 Miscellaneous Issues

We next discuss miscellaneous issues on Sign-to-911 design.

Compound Signs. In ASL, a compound sign combining two or more individual signs can be used to convey a single meaning. For instance, the *parents* sign is a compound one, by combining the signs for *mother* and *father*. By breaking down into their individ-

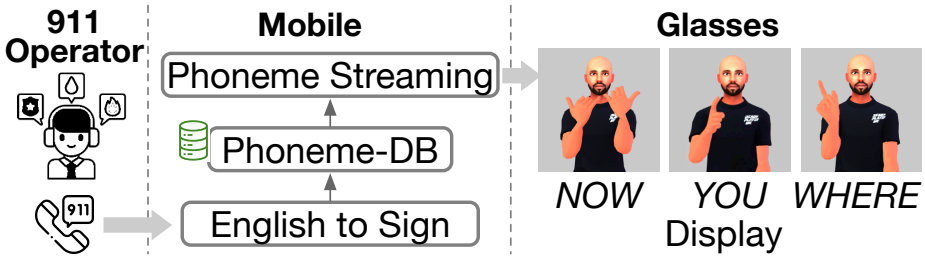

Figure 4.8: English to ASL Pipeline

ual components, we can efficiently handle such compound signs without increasing design complexity.

Voice/Text Interface. Most smartphones do not allow generating voices for phone calls directly from a mobile application due to security concerns. We thus send the translated English from the phone to the glasses. The AR glasses use their local text-to-speech to produce the voice, which is output to the speaker on the glasses. The glasses further play sounds to let the operator know that the user is signing. This way, the signer communicates with the 911 operator. Furthermore, we provide text captions for both the signer and operator's messages, which are displayed on the AR glasses for enhanced clarity and understanding.

Extreme Case Handling. In the extreme case the signer cannot make signs (e.g., got the hands/arm hurt in an accident), our system generates an automated, on-the-spot description. If no sign is detected after a 911 call, the glasses produce a voice message containing essential information, including name, age, and location, as well as surrounding objects/buildings learned from object recognition. This offers a critical lifeline for deaf or hard-of-hearing individuals during emergencies.

4.4 English to ASL Production

During an emergency call, an ASL signer also needs to comprehend voice responses from the 911 operator. Current speech recognition schemes [Cep] can readily convert the operator's voices into English texts. However, directly presenting such texts may hinder usability, as

many ASL users primarily communicate through the sign language rather than in English. To address this challenge, we construct an English-to-ASL pipeline (Figure 4.8). The pipeline generates signs from the operator's responses, animates the produced signs using a 3D avatar, and renders them on the AR glasses. This ensures that signers can directly understand the responses via signs, resulting in more effective call conversations.

There are two intuitive approaches to animate ASL signs on the glasses: one is to perform ASL translation and rendering at the smartphone, and stream the resulting animation video to the glasses; the other is to perform text-to-ASL translation and sign animation production on the glasses. However, neither of the two is desirable. The first approach incurs long delay due to encoding/decoding latency and prolonged transfer of video frames over the low-rate Bluetooth connectivity between the glasses and the phone. Whereas the second approach is too heavy to be operated on AR glasses with limited processing capacity and power budget.

We take a novel approach to lightweight production of ASL sign animations on AR glasses. To this end, we exploit the well-accepted MOVE-HOLD model [LJ89] to describe the temporal units of signs, and achieve a higher degree of compression for sign animation.

Specifically, a HOLD state records a static gesture, while a MOVE state captures the transition between two HOLD states. An ASL sign can be reconstructed using one to five HOLD states (ASL phonemes). Each phoneme effectively takes a snapshot of all involved sign parameters at the time instant for a HOLD state. It can be acquired by taking snapshots from the sign parameters discussed in §4.3.1. These phonemes serve as "key frames" during sign production.

Consequently, we design a three-step pipeline for ASL production based on phonemes (see Figure 4.8). The pipeline converts the incoming voice messages to phonemes on the smartphone, and streams the generated phonemes to the glasses for on-glass animation. The core of our pipeline is a module that produces ASL signs using phoneme sequences and compresses phonemes with kinematic correlations. Our scheme enables high-fidelity transfer of ASL signs via low-rate wireless connectivity (as low as 3.8KB/s).

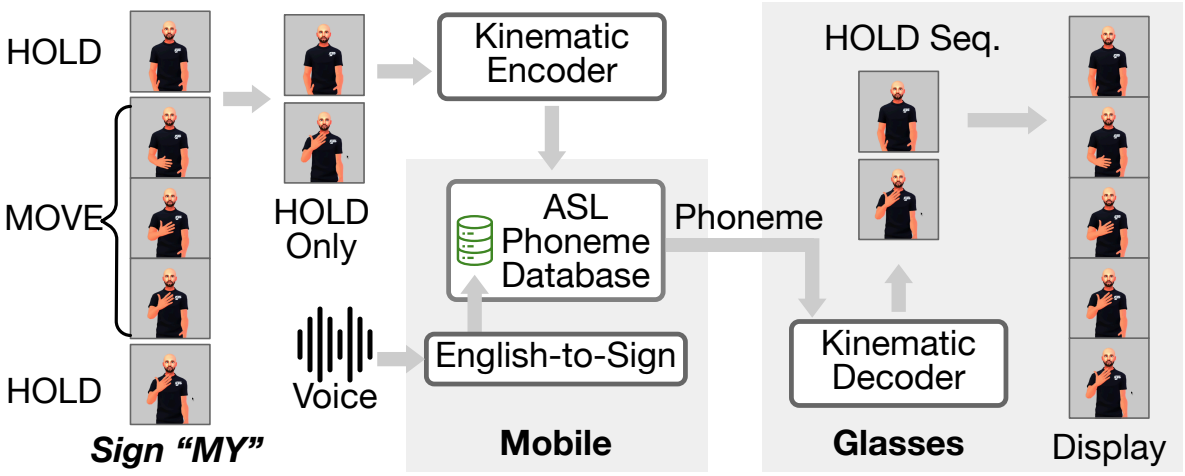

Figure 4.9: Phoneme DB construction & streaming

English-to-Sign. In the first step, incoming voices are converted to English sentences using the speech-to-text conversion module on the smartphone. We then translate the English text to ASL gloss (i.e., word sign) sequences. Note that ASL and English have different word orders. To ensure correctness of the produced gloss order, we reuse the syntax model in §4.3.3 and convert English sentences to corresponding gloss sequences. We match the English syntax order and ASL order, while dropping "be verbs" and articles. For example, the English sentence "What is your emergency" is translated as "YOUR EMERGENCY WHAT." Finally, we convert the gloss sequence to sign animations. Note that, each gloss must be matched by a sign of the same word type, since a gloss may assume multiple word types. For example, "SECOND" can be either an adjective or a noun, denoted with different sign gestures. Moreover, our animation database contains all frequently used signs. In case a gloss does not match any sign in the database, we create the fingerspelling animations from individual letters of the word.

Sign Production with Phoneme Sequences. We use the phoneme sequences to visually produce each recognized gloss. Direct usage of sign parameters of §4.3 does not yield an efficient method for producing a sign. This is because sign parameters record complete hand skeleton movements over time. Transmitting this entire skeleton sequence to the glasses

results in prolonged latency. In contrast, phonemes offer a more concise way to convey visual information for a sign. Phonemes follow the MOVE-HOLD model [LJ89], which includes HOLD states. A phoneme captures a static snapshot of the sign parameters at a particular timestamp. To capture these HOLD states, we first use body and hand recognition algorithms [BGR20, ZBV20] to extract the 3D skeleton sequence from sign language videos [LRY20]. We then identify HOLD states in the sequence based on movement speed and direction, which are verified by human experts to ensure accuracy. With these phonemes at HOLD states, our sign rendering uses spherical interpolation algorithms [Pen98] to approximate the MOVE state between two HOLD states.

Compression with Kinematic Correlations. We further compress phonemes by leveraging kinematic correlations. Note that, hand joint movements must comply with human kinematic constraints [ES03]. For instance, a single flexor tendon can control the flexion of multiple joints, such as the dip joint and pip joint [MI94]. With the kinematic constraints, each hand could be represented with a subset of 10 out of the full 15 hand joints. The degree of freedom of each joint rotation could also be reduced from 3 to 2 or 1 (depending on specific finger joints). We build a kinematic encoder/decoder with a Linear Regression (LR) algorithm. Only 10 joints and corresponding reduced rotations need to be transmitted during rendering. The remaining joints and rotations are inferred from the LR at the glasses. Our approach retains production accuracy under the limited bandwidth.

Figure 4.9 illustrates the construction and usage of the ASL phoneme database for phoneme streaming. We first extract and compress the HOLD states of each sign using the kinematic encoder, which is then stored in a local database at the smartphone. During translation, the phone looks up each gloss in the phoneme database and streams its compressed version to glasses. The AR glasses decode the body/hand movements with the LR and produce sign motion sequences by interpolating the phonemes of HOLD states [Pen98]. Finally, the glasses render the signs with a local 3D avatar in real time. The approach reduces the required bandwidth by 50x compared with directly streaming hand skeleton sequences,

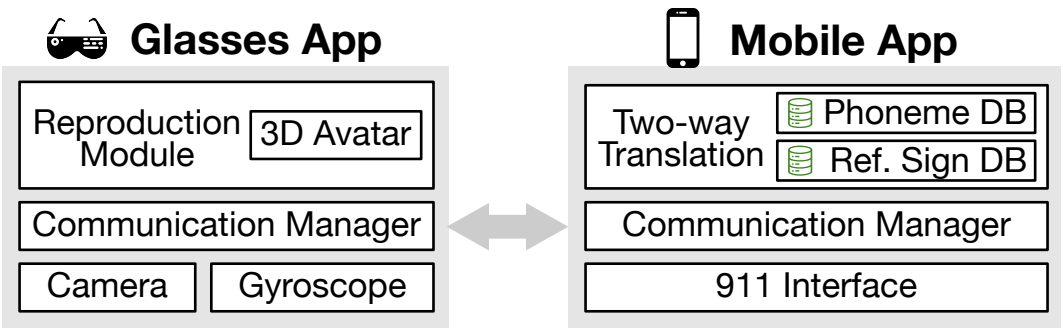

Figure 4.10: Sign-to-911 Implementation

ensuring both low latency and high fidelity.

4.5 System Implementation

The implementation of Sign-to-911 is in software only. It operates as apps on the phone and the AR glasses, respectively. The main components are shown in Figure 4.10.

Application on AR glasses. The application on glasses runs as an Android app on Android Go Edition [And23b], a lightweight Android on assistive AR glasses. The app includes 2824 lines of Java and 1149 lines of Kotlin. For ASL-to-English, the app acquires real-time camera views and gyroscope data with Android APIs [And23a, And23d]. The glasses record camera views into *H.264* videos at 15 FPS, which are later decoded by the smartphone for sign translation. Both encoding and decoding use Android MediaCodec [And23c]. For ASL reproduction, the glasses receive encoded ASL phonemes through the communication manager, and reproduce signs using a reproduction module. This module loads a 3D avatar from local files produced by Mixamo [Wik22]. To reduce rendering overhead, we downsize the texture and decimate polygons to create an avatar with 66 bones, 132 joints, and 28,106 triangles baked as a 2.4MB file in .glb format. The reproduction module animates this avatar with skeleton transformation and renders it with Google Filament [Goo22].

The AR glass uses a Communication Manager module (550 lines of Java) to enable glass-phone data communications. We use Bluetooth RFCOMM [ANO01] for reliable data

transfer. We have designed packet abstraction on top of the RFCOMM stream. The abstraction has two types of packets: control and data. Control packets coordinate between the AR glasses and the phone during 911 call initiation or termination. Data packets may carry three types of information: video frames, gyroscope data, and ASL phonemes. This communication manager is also reused by the smartphone app.

Smartphone App. We have developed an Android application for smartphones. This app has 4304 lines of Java and 2076 lines of Python code. The communication manager receives ASL-to-English data from the glasses and streams English-to-ASL phonemes. We use Android-MediaPipe [Med23] to capture the skeleton and run it on mobile GPU. Our translation module performs bi-directional translations between ASL and English. It is implemented with Andronix [And], thus allowing Python code and libraries to run on phones with similar performance as native apps. To build our recognition and translation modules, we utilize scikit-learn [PVG11] for machine learning, and NLTK [BKL09] for natural language processing in Python. Both the reference sign database and the phoneme database are embedded and preloaded in the app.

Moreover, we have implemented a 911 interface to make emergency calls. To capture the audio stream from a phone call, system permission is typically required due to Android privilege management [Man]. To bypass this roadblock, we bound our mobile app with an Android accessibility service [Acc], thus enabling us to capture the audio stream through the *VOICE_RECOGNITION* audio source. Our app thus runs continuously as long as the service is activated, and provides swift response upon emergencies. The captured audio stream is processed by speech recognition using VOSK library [Cep]. For ASL-to-English, we call Android's built-in text-to-speech engine [And23e] to generate English voices from the text.

Communication Core We chose the Bluetooth RFCOMM [ANO01] as the baseline communication protocol between the AR glasses and the smartphone. RFCOMM provides reliable data transmissions, which guarantees the delivery of bi-directional translation results.

However, it is also stream-based. To segment different video frames or translation results and to process them independently, we added the abstraction of packets on top of the RFCOMM stream. Packets are categorized into control packets and data packets. Control packets are used for coordination between the AR glasses and the mobile phone when a 911 call is initiated or terminated. A data packet encapsulates data of any one of the three types: a single video frame, a series of ASL animations, and a translated English sentence. We also implemented an automatic and constant connection between the glasses and the mobile app. After they are paired and connected for the first time, either device's MAC address is saved in the permanent storage of the other. Then if either app is closed and restarted, the MAC address of the other can be retrieved to re-establish the connection.

To facilitate the streaming of the camera preview from AR glasses to mobile phones, it is necessary to encode the raw video stream input into a more compact format to accommodate the Bluetooth bandwidth limitations. In our preliminary experiment, we observed that the stable bandwidth of Bluetooth typically reaches 150 KB/s under daily conditions. Hence, we have opted to utilize Android's MediaCodec for encoding the camera-captured video stream, specifically in the *YUV420SP* format, into the widely supported *H.264* format. We have set the frame rate to 15 frames per second and an I-frame interval of 1 per second.

Based on our experimental findings, the average encoded frame size amounts to a mere 5 KB. Consequently, the streaming process necessitates only 75 KB/s of bandwidth. On the mobile side, we have employed MediaCodec once again, this time for decoding the received *H.264* video stream back to the default *NV12* format. Furthermore, we have transformed the underlying bit arrangement to match the *NV21* format and subsequently passed the resulting byte array to the hand recognition model.

By employing this streamlined approach to encoding and decoding video streams, we ensure efficient utilization of the Bluetooth bandwidth while maintaining the necessary video quality for accurate hand recognition and subsequent processing on the mobile device.

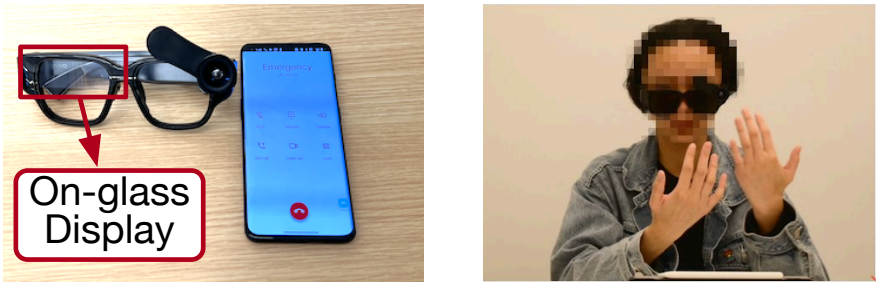

Figure 4.11: Experimental Setup

911 Call Interface According to [Man], unless the app is able to acquire system permissions, the audio stream of a phone call cannot be captured directly. Therefore, we bound the mobile app with an accessibility service of Android [Acc], through which the audio stream could be captured through the *VOICE_RECOGNITION* audio source. Moreover, after binding with an accessibility service, the app keeps running as long as the service is turned on, corroborating the idea of constant connection. After the audio stream bytes are captured, they are used for speech recognition through the VOSK library [Cep]. On the AR glasses, after the English texts translated from ASL are sent back, we use the default text-to-speech engine to generate voices from them.

4.6 Evaluation

We next evaluate Sign-to-911 on commodity smartphones and assistive AR glasses. We first introduce our evaluation methodology. We describe our model training results and assess both ASL translation and production. We further conduct a user study on signer's experiences.

4.6.1 Methodology

Experimental Setup We evaluate Sign-to-911 on off-the-shelf AR glasses and smartphones, as shown in Figure 7.5. Specifically, we use INMO AIR [INM23] assistive AR glasses running Android 10 Go. The glasses are equipped with a quad-core Cortex-A53 processor

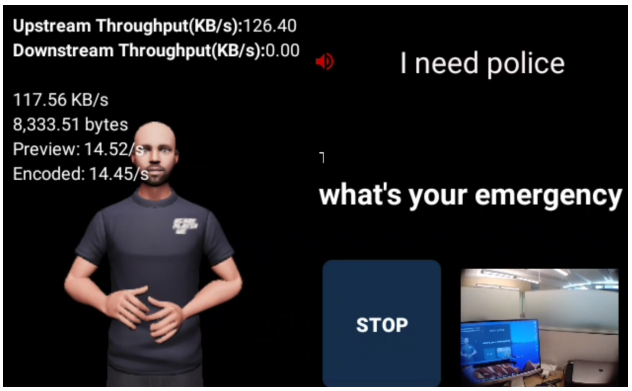

Figure 4.12: Glass Application

(1.4GHz), 2 GB RAM, and 32 GB flash memory. The smartphone, OnePlus 10 Pro 5G, runs Android 12 with a Qualcomm SM8450 Snapdragon 8 Gen 1 processor, 8GB RAM, and 256G storage. We further use an Eversame USB Digital Tester to gauge power consumption.

Glass-view Sign Traces. Our ASL-to-English model training requires: 1) ASL video and gyroscope data captured from the AR glasses; 2) the sign sequences and corresponding English text translations. However, there are no such public traces to the best of our knowledge. Consequently, we decide to collect our own glass-view ASL traces.

As our solution is designed for emergency call situations, we use two text-based 911 content sources to generate glass-view traces. The first from Montgomery County [CHI] contains over 600K emergency call records, each of which offers a summary report for the 911 call. The second is the 911 response template from Eugene Police Department [Dep]. It contains 46 questions covering 4 emergency scenarios (medical, fire, police, and hybrid). 911 operators use it to quickly understand the situation by asking appropriate questions.

We invite signers to drive ASL conversations in these two datasets. To ensure accurate representation of natural signing habits and experiences, we select a group of six signers in our study. This group includes two native ASL signers, an ASL linguistic expert, and three ASL students with over two years of signing experiences. The signers sign the sentences, and their signing motions and gyroscope readings are collected by two types of devices: AR glasses [INM23] and a head-mounted action camera [Ins23]. We label the collected data with

the sign sequences and corresponding English sentences.

The above collection process results in three glass-view ASL datasets, covering 249 GB of video traces and more than 11.5-hour user-signing actions. Dataset-1 (D1) logs 478 distinct ASL word signs using the first source [CHI] to cover the 911-related sign words. Dataset-2 (D2) is built from the 911 response template [Dep]. For each question, multiple answers are generated to cover various real-world scenarios. In total, we generate 180 Q&As, which use 202 distinct ASL signs. All questions and answers are validated by signers.

D1 and D2 together cover more than 550 distinct word signs used in 911 emergency calls. In addition to word signs, the remaining contents, e.g., names, addresses, and numbers, are recorded with fingerspelling motions. We use D1+D2 (i.e., the superset by merging D1 and D2) to train the sign recognition model. We further use D2 to evaluate the grammar/syntax model in the 911 settings.

We construct Dataset-3 (D3) to evaluate our overall system. D3 contains detailed 911 call conversations, both real ones and machine-generated ones. We download 4 real-life 911 call recordings [Ope20b, Ope20a] and translate these conversations into ASL. Furthermore, we use ChatGPT-3.5 [Ope22] to generate 30 artificial conversations with 150 Q&As for emergency calls on three topics (medical, fire, and police). All conversation contents are verified by our signers. To our knowledge, we have produced the largest glass-view ASL dataset to date.

Phoneme Database for ASL Production. Different from the glass-view sign traces, the phoneme database requires front views of signs, just the opposite to the glass views. We construct this database by extracting phonemes from the public ASL video datasets [LRY20]. For each sign, we label its HOLD-state frames, extract phonemes, and compress them using kinematic correlations. Our database covers 3100 signs, larger than those provided by ASL directory [TGB10] and linguistic study [SCC21a]. We collect more compound signs, while prior efforts [TGB10, SCC21a] did not. The words not covered by these signs are expressed

through fingerspelling.

Ethical Considerations. This research was approved by the Institutional Review Board of a large public university. We provided information sheets to all the participants, and their confidentiality and privacy were ensured. The user study followed ethical guidelines outlined by our institute; the ethical conduct of the study was in accordance with the principles of research ethics. Moreover, the used glass-view camera captured the sign motions view without any facial features of a signer. Furthermore, no actual 911 calls were made during this study. In summary, our research was carried out in an ethical and responsible manner.

4.6.2 Component Evaluation

In this section, we evaluate three main components of Sign-to-911: sign recognition model, grammar/syntax model, and English-to-ASL production.

Sign Recognition Model. We use the combined datasets of D1 and D2 to train the sign recognition model. The combined dataset D1+D2 is split into training (80%) and test subsets (20%). We evaluate our model training using 5-fold cross-validation on the training subset. The training yields an average accuracy of 91.72%, with a standard deviation of 1.27% over the training subset.

We compare the accuracy with two state-of-the-art recognition models, I3D [LRY20] and SL-GCN [JSW21]. Both take the image view as input. For fair comparisons, these two models are also trained with the same training data subset as ours.

Model	**Accuracy (%)**
I3D	82.03
SL-GCN	89.06
w/o-cat.	85.72
Sign-to-911	88.53

Table 4.2: Sign Recog. Acc.

The accuracy results are shown in Table 4.2. It is clear our model has comparable

accuracy in sign recognition as the SL-GCN model, but higher accuracy than the I3D model. The root cause analysis shows that, the inaccuracy of our model largely stems from those signs that are prone to sliding to the view boundary or outside the camera frame. For example, the signs for the words BROWN and RED are close to the face, thus being on the view boundary. Consequently, the hand-tracking algorithm fails to accurately detect the skeleton for both signs. Minor adjustments to the camera view angle of the AR glasses may solve the problem.

Note that, we only compare our model with the vision-based solutions. Other sensor-based schemes are trained and tested using spatial-temporal sensory data, and cannot be trained using our visual datasets. Moreover, they have only tried with small datasets of 50-100 signs [FCZ17, HLZ19], while our model covers about 550 distinct signs (4.5× improvements).

Syntax Model. We train our syntax model using the public dataset [Sig23b] with the approach described in §4.3.3. To evaluate the quality of syntax translation, we use D2 that has both English and ASL conversations. We evaluate the syntax model for both ASL-to-English and English-to-ASL translations, using the word accuracy metric (i.e., WAcc [Koe09]). In ASL-to-English translation, we convert the gloss sequences using the syntax model, and compare them with the corresponding English sentence. In English-to-ASL translation, we convert the English text to the gloss sequence, and compare the obtained gloss sequence with the ground-truth one.

The evaluation results show that, the WAcc for our ASL-to-English translation is 93.96%, and the WAcc for English-to-ASL is 95.60%. The errors in ASL-to-English translation were primarily due to difficulties in filling in articles (a/the), while the errors in English-to-ASL translation were due to differences in word order in ASL, which did not affect the meaning of the translation. Overall, our syntax model shows high accuracy in mutual translations between ASL and English.

ASL Production. We compare the quality of ASL production based on video streaming and that using our glass-based animation. Figure 4.13 shows the visual effects produced under 10KB/s video, 30KB/s video, and our on-glass production. The video-based production results in poor video quality, making it difficult to recognize signs. In contrast, phoneme streaming and on-glass production achieve high-fidelity production with low-rate, mobile-glass communication.

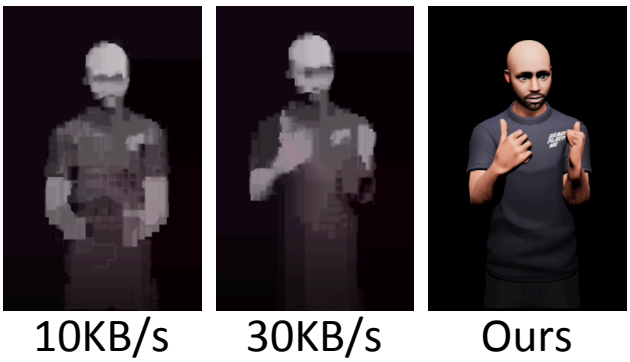

Figure 4.13: Visual Quality

To quantify the consumed bandwidth, we compare three streaming methods: full skeleton sequence streaming, phoneme streaming with/without kinematic compression. Experiments show that, phoneme streaming consumes the least bandwidth by transmitting only the essential HOLD states. Full skeleton sequence streaming uses 78.6KB/s bandwidth, while phoneme streaming without kinematic compression only consumes 11.2KB/s. With kinematic compression, the used bandwidth is reduced by 20.7× to merely 3.8KB/s.

ASL Animation Quality. We assess the quality of ASL animations by examining their recognizability, the accuracy of their parameters, and the hand distance between the human signer and the avatar. Firstly, our research survey reveals that all participants successfully identified the sign animation sentences in D3, despite their individual preferences for certain signing variants. Secondly, a comparison of the sign parameters with the ground truth reveals that a significant 95% were accurately represented in the animation, according to expert feedback. Finally, the average difference in hand position between the generated signs and

the reference human signs (ground truth) is less than 4cm. This discrepancy largely resulted from variances in body shapes between the human signer and the avatar. In conclusion, our animations successfully emulate key movements and sign parameters, with room for improvement in the areas of fluidity and reference accuracy. These aspects can be enhanced by integrating more human feedback and upgrading collision detection mechanisms.

4.6.3 System Evaluation

Sentence Translation. We first assess sentence translation in both accuracy and robustness. We use word accuracy (WAcc) as the metric for accuracy, which is calculated for both gloss sequences and English sentences. We compare ours with I3D and SL-GCN. Since I3D and SL-GCN do not support segmentation and syntax translation, we apply the same scheme as ours in these steps. All models are trained with the combined D1+D2 set, and then evaluated on D3. Note that, D3 is never used in training and records real-world 911 conversations.

Model	**D3**	**New Signer**
I3D	80.18	37.49
SL-GCN	86.31	56.32
Sign-to-911	**91.37**	**82.40**

Table 4.3: Word Accuracy (%) for Sentence Translation

Table 4.3 lists the comparison between our approach and the two related models. Sign-to-911 yields 5.06% higher in WAcc than SL-GCN, and 11.19% higher than I3D. Our detailed analysis reveals that the context model indeed helps; it improves WAcc from 84.49% to 91.37%.

We also evaluate translation robustness with respect to new signers. We collect the sign videos from a new signer under emergency call settings. The signer has not appeared in the training datasets. Results show that Sign-to-911 still achieves 82.40% accuracy (shown in

Table 4.3). Both sign parameters and categorization help to reduce the impact of varying sign habits and speeds. In contrast, I3D and SL-GCN barely achieve 37.50% and 56.32% accuracy, respectively.

Robustness Upon Emergencies Emergency situations can have more uncertainties than normal conditions. Diverse signer and environmental factors can result in varied data capture, which may consequently degrade sign recognition accuracy. It is thus necessary for our solution to accommodate these conditions. We utilized the same pre-trained model to assess recognition performance in different settings, including fast-paced walking at 4 miles per hour, low-light environments with illumination of 40 Lux, and outdoors. Across all settings, we evaluated recognition performance on the same subset of sign interactions from D3. Table 4.4 shows the results of comparison against the baseline lab environment with the illumination of 500 Lux. The recognition accuracies fall within the ±3%, indicating that our solution is robust enough to handle broad signer and environmental conditions.

Condition	**Accuracy(%)**
Lab	90.7
Low-light	92.8
Walking	90.9
Outdoor	88.9

Table 4.4: Accuracy Under Different Environment

End-to-End Latency. We quantify the end-to-end latency for both ASL-to-English and English-to-ASL pipelines. For ASL-to-English, we measure the interval from when a sign is completed to when the corresponding voice is generated. We define translation latency as the time it takes for the model to produce a translation. The remaining components (video encoding/decoding, transmission, etc.) are collectively referred to as capturing latency. We compare the latency with SL-GCN model on mobile (Mobile-SG+) and the cloud (Cloud-SG). For Cloud-SG, the phone transmits the video to the cloud through 5G for translation. The cloud runs the SL-GCN model with an Nvidia RTX 3090 GPU. The SL-GCN model running on mobile takes an average of 62 seconds due to its slow skeleton extraction module.

Alternatively, we extract the skeletons with [Med23] and feed them to the model. This modified version is referred to as Mobile-SG+. We also gauge our model without sign categorization.

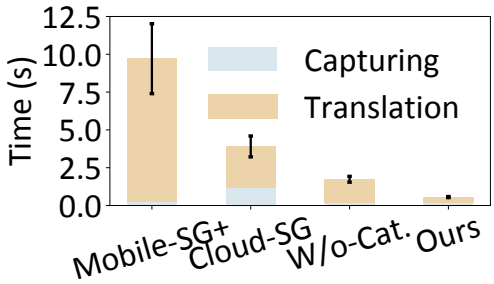

Figure 4.14: A2E Latency

The results are shown in Figure 4.14. Our solution achieves an average time of 0.55 seconds, resulting in 17.4× reduction for Mobile-SG+ (9.7s). Compared to directly running SL-GCN on the mobile, our solution reduces 112× of latency. Although Cloud-SG reduces processing time with its powerful GPU, it still takes 7× longer than our solution. Compared with the pipeline without sign categorization, our solution reduces translation latency by 8×, and achieves 3.1× reduction in end-to-end latency. Therefore, Sign-to-911 provides swift ASL-to-English translation on commodity mobile devices.

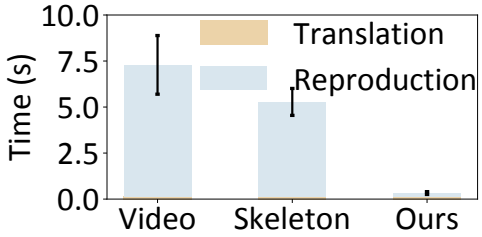

Figure 4.15: E2A Latency

We quantify the English-to-ASL latency from when the voice is received to when the glasses render the sign. As shown in Figure 4.15, we compare three solutions: streaming generated video at 480P, streaming full skeleton sequences, and our phoneme streaming. The translation on mobile takes 122ms on average, and phoneme streaming only takes 206ms to render the animation. In contrast, video-based and skeleton-based animations incur 22× and

16× latency, respectively. Our phoneme streaming thus ensures low-latency sign production despite using Bluetooth.

Model Complexity & System Overhead. The deep learning models for sign recognition use millions of parameters in their neural networks [LRY20, JSW21]. We use traditional AI/ML models with domain knowledge with only ∼4,000 parameters to be trained in our two-way, ASL-English communications. It significantly prunes the search space with ASL domain knowledge and offers lightweight solution.

We further show the applicability of our solution on mobile phones with different hardware capabilities. In addition to OnePlus 10 Pro 5G (∼550 USD), we run Sign-to-911 on Mate 20 (<300 USD) that runs Android 10 with a Kirin 980 processor, 6GB RAM, and 128GB storage. Results show that Sign-to-911 works with both high-end and mid-range smartphones. The average translation latency is 650ms, still a 14.9× reduction from Mobile-SG+.

We measure the system overhead by recording the power, CPU, and memory consumption over time during the operation of the entire setup on each device. Our system's power consumption is comparable to that of other camera applications or media players, less than 1.89W on glasses and less than 2.75W on smartphones. The mobile application uses less than 35% of CPU and takes up 350MB of memory, while the glass application uses less than 50% of CPU processing and takes up 65MB of memory. These results suggest that our application is efficient and should not significantly drain device resources, making it compatible with a wide range of commodity devices.

4.6.4 User Study

To evaluate the performance of our system in real-world scenarios, we conduct a user study. For emergency calls in D3, we ask 12 participating signers to rate the quality of experience (QoE), in terms of accessibility, usability, and overall experience on a 5-point scale: Excellent

	Accs.	Usab.	Oval.
Text-based	2.9	2.8	2.8
VRS	3.4	3.5	3.6
Sign-to-911	**4.2**	**4.3**	**4.6**

Table 4.5: User Study

(5), Good (4), Fair (3), Poor (2), and Very Poor (1). We further consider the diversity of the signers' background. We conducted a survey among users aged 20 to 80, which included the deaf, hard-of-hearing individuals, and students learning ASL. The accessibility reflects how readily the system can be learned by new users and activated upon emergencies. The usability reflects correctness, liveliness, and human likeness; it is well accepted in translation assessment [SRT10]. Signers further rate their overall experiences. They also use and rate two other emergency call solutions: text-based and video relay services (VRS). As shown in Table 4.5, our solution achieves the best QoE on both accessibility (4.2) and usability (4.3). The overall experience is 4.6, which indicates improved user experience compared with the text-based scheme (2.8) and VRS (3.4). Through interactions with users, we discovered that opinions about text-based communication significantly vary, mainly because many deaf individuals are not yet familiar with the new Text-to-911 service [Com23]. The user feedback highlights the importance of having a single-click solution for deaf people to make emergency calls. It confirms the demand for accessible emergency call services by the deaf community.

4.7 Discussion

Signing Variants & Iconic Classifiers. Differing methods and techniques in Deaf education and contact with English over the years have led to a continuum between two signing varieties: American Sign Language (ASL) and Signed Exact English (SEE). Because individuals in the Deaf community sign along this continuum, it is critical to be aware of such diversity in order to better adapt the model to be an accessible tool for the Deaf community. In addition, we must take into account this continuum along with other varieties of ASL,

such as regional varieties and Black ASL (BASL), for future studies.

Some signs change the path movement in order to convey different semantic meanings. This usually applies to a limited number of signs that are iconic[2] in nature (e.g., FIRE, ACCIDENT). Nearly all documented sign languages, including ASL, utilize iconic constructions called classifiers, signs that "represent the position or movement of an entity in a highly iconic fashion[Sch21]." Classifier constructions in sign languages can function syntactically as predicates, critical to an adequate translation of a sentence's semantic meaning. Some signs, such as FIRE, can be expressed as classifier predicates in which the sign is modified in its production to convey various levels of fire. This iconicity of classifier predicates presents a challenge for sign language translation as signs that may be signed under a normal condition may differ in production (e.g., different path movement) from those signed under different conditions. Since the number of commonly used classifiers is limited [Vic23] and variants mainly come from changes in path movement, our solution has a high chance of covering sign variants with improvements from more diversified training samples.

Device Compatibility Our solution has been meticulously designed to align with the common design of assistive glasses, typically encompassing a camera and display. Such compatibility allows glasses with these components to be seamlessly integrated with our solution. While AR headsets could also be used, they might not be ideal for prolonged usage due to potential discomfort and inconvenience. This inherent flexibility of our solution enables its application across a vast spectrum of devices, extending its accessibility to a larger user demographic.

Sign Language Universality Our solution is not exclusive to American Sign Language (ASL), it also demonstrates applicability across various sign languages. Given the shared parameters across different sign languages, our methods of capturing, translating, and producing sign languages can be easily adapted. To apply our solution to other sign languages,

[2]An iconic sign is one that specifically denotes the visual characteristics, such as shape and size, of its referent [VL00].

minor modifications like creating new handshape references and updating the grammar model might be needed. These adaptations potentially broaden the scope of our solution across a wide array of sign languages.

Extensive Use Case Applications The utility of our solution extends beyond emergency calls, demonstrating potential applicability across a diverse range of use cases. We envisage that the glass-based ASL translation can be effectively employed in different forms of communication, including phone calls and direct interactions such as customer service encounters, grocery shopping, and daily conversations. Our future development plans include expanding the sign language coverage and incorporating complex grammars like ASL classifiers to further enhance the versatility and functionality of our solution. This advancement has the potential to drastically improve accessibility, facilitating communication for individuals who primarily rely on sign language.

4.8 Related Work

Sign Language Translation. Sign language recognition has been an active research topic in recent years. The existing solutions are either vision-based (I3D [LRY20], SAM-SLR [JSW21], and DeepASL [FCZ17]) or sensor-based (using gloves [AZZ18], smart watches [HLZ19], earphones [JGZ21], EMG sensors [ZJW22], etc.). These solutions, regardless of how to extract sign features, apply deep-learning-based models (e.g., RNN and its variants) on temporal data for sign recognition. Given the model complexity, the trained model could only cover a vocabulary set size of around 100 signs. In contrast, our solution departs from the deep learning based approach. We use simpler, traditional AI/ML models, while exploiting rich ASL domain knowledge. We thus recognize about 550 signs. Our solution runs on commodity smartphones for recognition without cloud/edge support. We further reduce translation latency by an order of magnitude. Previous research [SSB20] employs the Hamburg Notation System (HNS) [Han04] for teaching sign language, but its complex combinational represen-

tation and lack of design for recognition prompted us to use the simpler Stokoe system in this paper.

Moreover, existing solutions cannot well handle sentence-level translations for ASL. They either offer no sentence-level translation at all [LRY20, JSW21], or rely on complex temporal model training on large sentence corpora [FCZ17, AZZ18, HLZ19, JGZ21, ZJW22]. In contrast, we explicitly consider ASL grammar and embed such domain knowledge into our sentence-level translation models. In summary, we pursue an explainable AI model approach, thus departing from the blackbox deep learning schemes. Our models are simpler and run on smartphones.

Sign Language Production. For sign language production, current apps, such as HandTalk [han23] and Sign Language Translator [Sig23a], cannot often produce the correct grammar order. They also cover limited signs, rely on clouds/edges for sign production [Viv23, ASL23], or require heavy GPU processing for rendering and generation [SCH20, SHB20, SCB20]. In contrast, we devise a grammar model using ASL syntax knowledge, thus providing both accurate and high-fidelity ASL reproduction. Our solution runs on mobile devices, and renders signs and ASL sentences on AR glasses.

Assistive AR/VR. AR/VR technologies have been used to develop assistive applications for people with disabilities [Dic21]. For instance, [ZBB18] and [BH20] enable navigation for people with visual impairments, [VOI23] leverages AR for social and emotional learning, etc. We report the first system using assistive AR glasses to support ASL communications. Our software-based solution on commodity AR glasses makes it readily accessible to ASL users.

4.9 Conclusion

As we draw this chapter to a close, we emphasize the vital importance of sign language support for individuals with hearing disabilities, particularly in emergency scenarios. In today's high-speed, interconnected society, a significant number of people depend on sign

languages like ASL as their primary means of communication. The urgency of this need escalates in emergency situations, where instant, dependable, and accessible communication channels are a necessity. Regrettably, many of the current solutions fall short, being deemed inconvenient or, in some cases, inaccessible.

In response to this pressing issue, this chapter has introduced our solution, Sign-to-911. This rapid and lightweight system has been designed specifically to bridge the gap in ASL-English communication during emergencies. Sign-to-911 effectively employs traditional AI/ML models but also integrates specific ASL linguistic domain knowledge, allowing us to reduce model complexity while retaining high accuracy and speed in translations. The system's effectiveness has been validated through rigorous evaluations with real signers.

Sign-to-911 was subjected to a rigorous stress test using the 911 call service. This service not only requires high recognition accuracy but also necessitates swift bidirectional translations. Importantly, all of these requirements are met using mobile and wearable devices, without the need for cloud/edge support.

To sum up, Sign-to-911 offers a swift, lightweight, and accessible solution, addressing the urgent need for effective ASL-English communication in emergency scenarios. We are confident that by offering a more accessible and efficient solution for emergency communication, we are opening a pathway to potentially save lives in critical moments. The core value of Sign-to-911 is its ability to meet the immediate and significant needs in emergency communication, making it an essential tool in the ASL user community.

CHAPTER 5

Advancing Sign Language Translation in Accessibility Solutions

The previous chapter introduces our approach to sign-oral language translation for emergency communication. In this chapter, we discuss our attempts to bridge the communication gap in general usage cases. Compounding these challenges is the notable dearth of suitable training datasets, a significant obstacle in the effective training and development of machine learning models, especially for assistive AR.

The organization of this chapter is as follows: We commence with §5.1, where we discuss the pressing need and the significant challenges associated with general sign language interaction. In §5.2, we introduce a novel method for sign recognition that uses non-video references. §5.3 presents this approach's analysis and implementation.

5.1 Sign Language in Accessibility Services

5.1.1 Accessibility for Deaf and Hard-of-Hearing

Accessibility standards have been established in various regions around the world. These include the Web Content Accessibility Guidelines (WCAG) by the World Wide Web Consortium (W3C) [Con18], the Americans with Disabilities Act (ADA) in the United States [ada90], the European Accessibility Act (EAA) within the European Union [Uni23], and the Accessibility for Ontarians with Disabilities Act (AODA) in Ontario, Canada [aod05].

Ensuring accessibility for the deaf and hard-of-hearing community means providing accommodations such as sign language interpretation videos and closed captioning for multimedia content. Additionally, for public services the ADA requires businesses and nonprofits that serve the public to make "reasonable modifications" to ensure equal access for people with disabilities. This includes providing accommodations such as sign language interpreters in hospitals for the deaf and hard of hearing. Without these accommodations, many people may struggle to access information, participate in activities, or benefit from digital advancements. By being inclusive, these accessibility standards not only protect the rights of the deaf community but also contribute to the diversity of interactions and experiences.

Despite the emphasis on accessibility for the deaf and hard of hearing, the practical implementation of these services frequently encounters significant challenges. The presence of ASL interpreters is often inconsistent [Cen23], and the online content's sign language interface is insufficient. Due to these shortcomings, many from the deaf community find themselves marginalized, facing barriers to quality education, online engagement, and daily routines.

5.1.2 Challenges in Delivering Sign Language Services

Effective machine sign language translation is a complex and demanding task, impeded by numerous hurdles, primarily rooted in the inherently distinctive characteristics of sign language. Unlike conventional spoken languages, sign languages employ a dynamic three-dimensional motion and visual context, requiring specialized strategies for machine learning model training and translation methodologies. This section outlines the primary challenges encountered in the development and implementation of effective and generalizable sign language translation algorithms.

Insufficiency of Labeled Video Training Sets for Signs A prominent challenge in machine learning applications for sign language translation pertains to the scarcity of ad-

equately labeled video datasets for signs. The success and efficiency of machine learning models are contingent on their training with comprehensive, diverse, and accurately labeled datasets. For sign language translation, this translates into the need for high-quality video footage that captures a wide array of sign language vocabularies executed by different signers. Subsequently, each sign must be accurately labeled, a process that necessitates expert knowledge of sign language, alongside significant time and effort. The limited availability of such precisely labeled video training sets imposes considerable constraints on the robustness and accuracy of translation models, hindering the progression of sign language translation technology.

Shortage of Parallel ASL Datasets Efficient sign language translation faces another major obstacle - the lack of parallel American Sign Language (ASL) datasets. In machine translation, parallel datasets that contain sentence pairs in two languages with equivalent meanings are crucial for training translation models. These datasets enable the mapping of meanings between two different languages, which is essential for achieving effective translation.

While ample parallel datasets exist for various spoken languages, generating equivalent datasets for ASL and English involves unique challenges. The primary hurdles include the complex task of recording synchronous ASL and spoken English translations, reconciling the different grammatical structures, and accommodating the visual-spatial nature of ASL in contrast to the linear nature of spoken English. This acute shortage of parallel ASL datasets presents considerable obstacles to the development of high-performing machine translation models, thereby impeding progress in ASL-English translation.

To address these dual challenges, we unveil our innovative approach designed to facilitate the learning of the translation pipeline despite a limited training dataset. Our methodology encompasses two key components: first, we leverage dictionary-based descriptions to learn the representation of ASL signs, thereby bypassing the need for extensive labeled video data.

Secondly, we employ a learning-based approach to understanding ASL grammar, drawing on a set of predefined, constrained rules. By strategically maneuvering within the boundaries of these limitations, we endeavor to optimize the capabilities of our translation model.

5.2 Sign Recognition with Non-Video Reference

Approaches [LRY20, RKE21b] applied to camera-based sign language recognition predominantly center on using video samples as the primary resource for learning the sign recognition model. This intuitive strategy, while having its merits, grapples with significant constraints that limit its overall efficacy and scalability. The first notable issue arises from the heavy dependence on data collection, requiring an immense repository of video samples from ASL users to ensure a robust learning process. Each sign demands multiple instances to prevent misinterpretation and errors, thereby multiplying the data collection effort. Additionally, a secondary challenge stems from the scalability of this method. Extending the scope to encompass a comprehensive catalog of signs presents logistical complications, effectively hindering the adoption of this approach on a larger scale.

In the face of these formidable challenges, we have embarked on a quest for an innovative, effective solution. In our quest, we drew an intriguing parallel between ASL and spoken languages, particularly regarding representation. Spoken languages have successfully adopted the International Phonetic Alphabet (IPA) as a standardized written form of the pronunciations, enabling consistent and universal representation. Borrowing from this strategy, we hypothesized that sign language could also be effectively represented using a dictionary-like format. By systematically correlating these dictionary entries with their corresponding physical manifestations – the hand/body movements – we established an unconventional yet promising method for learning signs without a mandatory reliance on video samples.

5.2.1 Insight from International Phonetic Alphabet

Our strategy was further bolstered by the exhaustive lexical descriptions provided by ASL-LEX [CSC17, SCC21b]. This resource served as a valuable bridge, allowing us to translate the lexical descriptions into identifiable movement features. Following an in-depth analysis of the descriptions, we delineated a distinct set of features that effectively encapsulated the range of hand and body movements required to perform each sign. By establishing this translation from lexical descriptions to movement features, we crafted a robust understanding of ASL with the descriptions.

The key insight underscoring our approach is that sign language, much like its spoken counterparts, can be represented in a dictionary format. This innovative perspective allows us to bridge the gap between dictionary entries and physical hand/body movements, enabling the learning of signs devoid of video representations and thus bypassing the limitations of conventional video-based learning approaches.

5.3 Analysis and Implementation

The initial step in offering dictionary-based translation involves constructing a dictionary detailing signs and their descriptive parameters. We harness three resources for this endeavor: ASL-LEX [SCC21b], which provides tables of lexical features; SigningSavvy [Sig23b], which offers text descriptions; and the American Sign Language Handshape Dictionary [TGB10], a conventional dictionary enumerating sign parameters. By incorporating features from these sources, we have assembled a sign-parameter database. This database comprises 7 categories of discrete parameters—including hand number, handshape, same/different handshape distinction, major location, minor location, repetition, and movement shape—and presents between 2 to 58 candidates for each parameter.

To enhance our understanding of sign recognition, we delved deeper into the lexical parameters extracted. Using a comprehensive chart for Sign Identification Using the Dictionary,

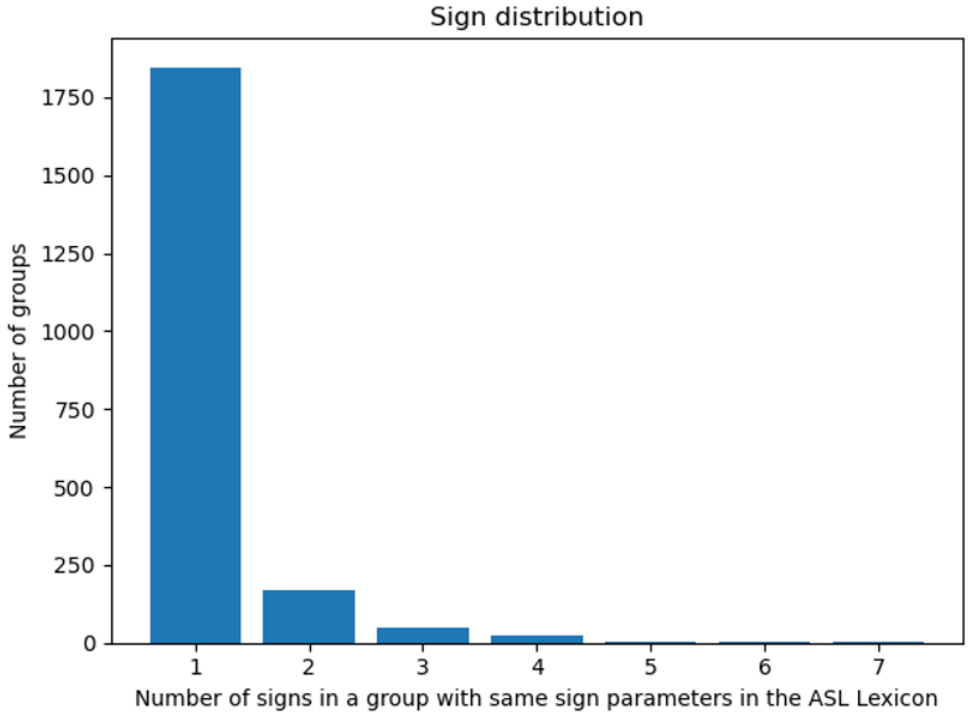

Figure 5.1: Distribution of Sign Lexicon Groups

we noted the efficacy of our database. From a single source, only 56.0% of the signs can be uniquely discerned based on the provided parameters. However, after the concatenation of our diverse sources, the accuracy exhibited a marked improvement:

- 74.7% of the signs could be singularly pinpointed (Figure 5.1).
- We observed a 94.4% identification rate when considering the top-3 possible matches. Further expanding the scope to the top-5 possible matches yielded a remarkable 98.9% accuracy.

We utilized the AR/VR glasses to record these parameters in real time. Currently, our system is implemented on the Unity AR platform (as illustrated in Figure 5.2), which is adept at tracking the motions of fingers, hands, and the head concerning stationary world objects.

Testing with the goggles shows that our approach is able to capture the

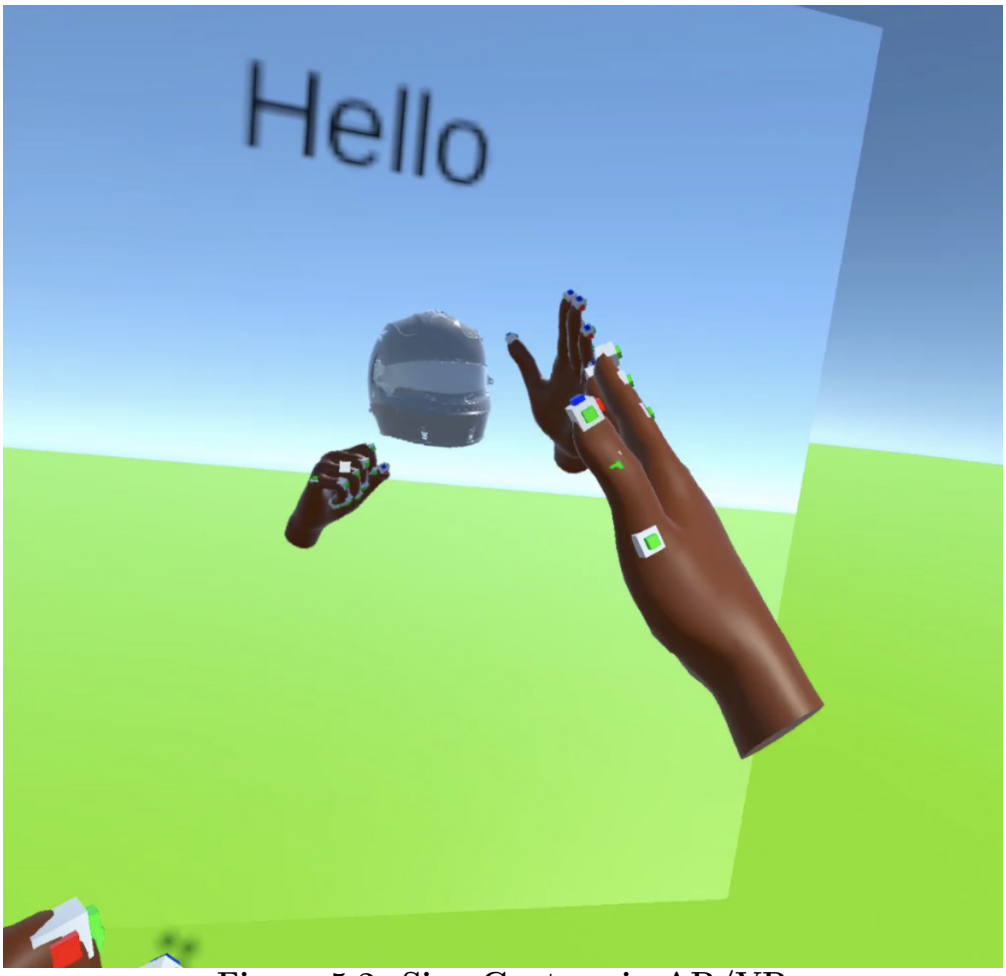

Figure 5.2: Sign Capture in AR/VR

5.4 Conclusion

This chapter delved into the pressing issues of accessibility services for the deaf community in general conditions. We have highlighted the existing gaps and the need for more comprehensive solutions. To address these challenges, we proposed a solution design that leverages a dictionary-based corpus for general sign language recognition. This design not only promises to bridge the accessibility gap but also offers a scalable and adaptable approach for various scenarios. Furthermore, we discussed the implementation and analysis of this solution, shedding light on its practicality and potential outcomes.

In the next chapter, we will introduce AnySign, our developed platform. It is designed to enhance interaction and data collection by leveraging a user-feedback loop, aligning with our dictionary-based design.

CHAPTER 6

AnySign: An Open Platform for Sign Language Interaction and Sign Documentation

In this chapter, we explore the development and structure of an integrated online platform aimed at enhancing the learning and interactive experience with ASL. The platform seeks to connect the Deaf community and the hearing population through a multifaceted approach to sign language translation and engagement.

In addressing the pressing challenges faced by users and learners of ASL, this innovative platform integrates four key modules: Sign Dictionary, English to ASL Translator, Teach-Me-Sign, and SignChat. Each module is crafted to alleviate specific issues, collectively providing a holistic solution that caters to the distinctive needs of the ASL community.

The *Sign Dictionary* module introduces a system that offers detailed animations and documentation for each sign. It supports not only ASL users but also aids ASL learners. The *English to ASL Translator* is designed to transform English input into ASL syntax and grammar, serving as an effective learning interface for both beginners and intermediate learners. *Teach-Me-Sign* leverages community contributions to address the issue of ASL variations. Finally, *SignChat*, a module that integrates sign language with advanced AI language models, propels the interpretation and generation of signs into the digital era.

The following sections will expound upon the approach and the unique ways this online platform has been designed to resolve the challenges intrinsic to ASL communication and learning. Further details and live interaction with the system can be accessed via our website

at `AnySign.net`. Through the presentation and discussion of this innovative platform, we hope to bring a new dimension to the world of sign language, making ASL more accessible, interactive, and integrative.

6.1 Overall Platform Design

The platform's overall structure is depicted in Figure 6.1. Sign dictionary, Teach-me-sign, and SignChat are three applications of our system, while the translator is achieved by our translation core functionalities. It is also used for other modules and mainly runs on our backend, so it is not listed in the applications.

The web system implementation is divided into two critical parts: the frontend and the backend. The frontend is responsible for the user-facing aspect of the web application, managing all user interactions, real-time animation interpretation, and webpage updates. It includes a sign production interface using Three.js[Cc23], a cross-browser JavaScript library, a MoCap module for video stream in Teach-me-sign and SignChat, and other pages or components. Users can access the platform through these interfaces and utilize the key modules.

On the other hand, the backend, built using the Flask framework, serves as the backbone of the system, responsible for handling all data processing, access to the algorithm library, and communication with the database. It can load our key algorithms and models in the translation-core, responsible for sign recognition and production, and the SignCorpus, designed for avatar transformation processing. Specific APIs are designed to complete interactions with the frontend.

The frontend and the backend communicate through Axios, a powerful and promise-based HTTP client for Node.js and the browser. The frontend makes use of Axios to send and receive requests to and from the backend. Then, the backend interacts with the database and processes data according to the requests received from the frontend. It may also receive

feedback from the frontend.

Our database, which includes two types of tables - one with sign-related information and another with user information for the user system - can be accessed by the backend. It can be modified by the backend order and provide the necessary data.

This architecture ensures that the web system is scalable, maintainable, and responsive to user interactions, enabling efficient communication between the frontend and backend and providing a seamless user experience.

6.1.1 Frontend

Interface layout The design of the user interface (UI) is a critical aspect in web and software development, as it greatly impacts the user's overall experience with the application. Our aim with the UI layout is to create a design that is easy to navigate and intuitive to use.

The website's interface layout consists of three main components: the navigation bar, page content, and footer. The navigation bar acts as the primary navigation tool, helping users navigate throughout the website with ease. It includes three key elements: the logo, menu, and user component. The logo, located on the left side of the navigation bar, provides a quick link to the home page, which also serves as the sign search page. The menu includes several buttons that direct users to different pages within the website. For instance, the "products" button contains a sub-menu that links to the website's three main offerings: the sign dictionary, the ASL sign translator, and the "Teach-Me-Sign" page. The menu also collapses automatically to ensure a consistent user experience across different devices and screen sizes. The user component, located on the right side of the navigation bar, displays either a login button when the user is logged out or the user's avatar when logged in or logged out.

The page content can be filled with various components and is designed based on user

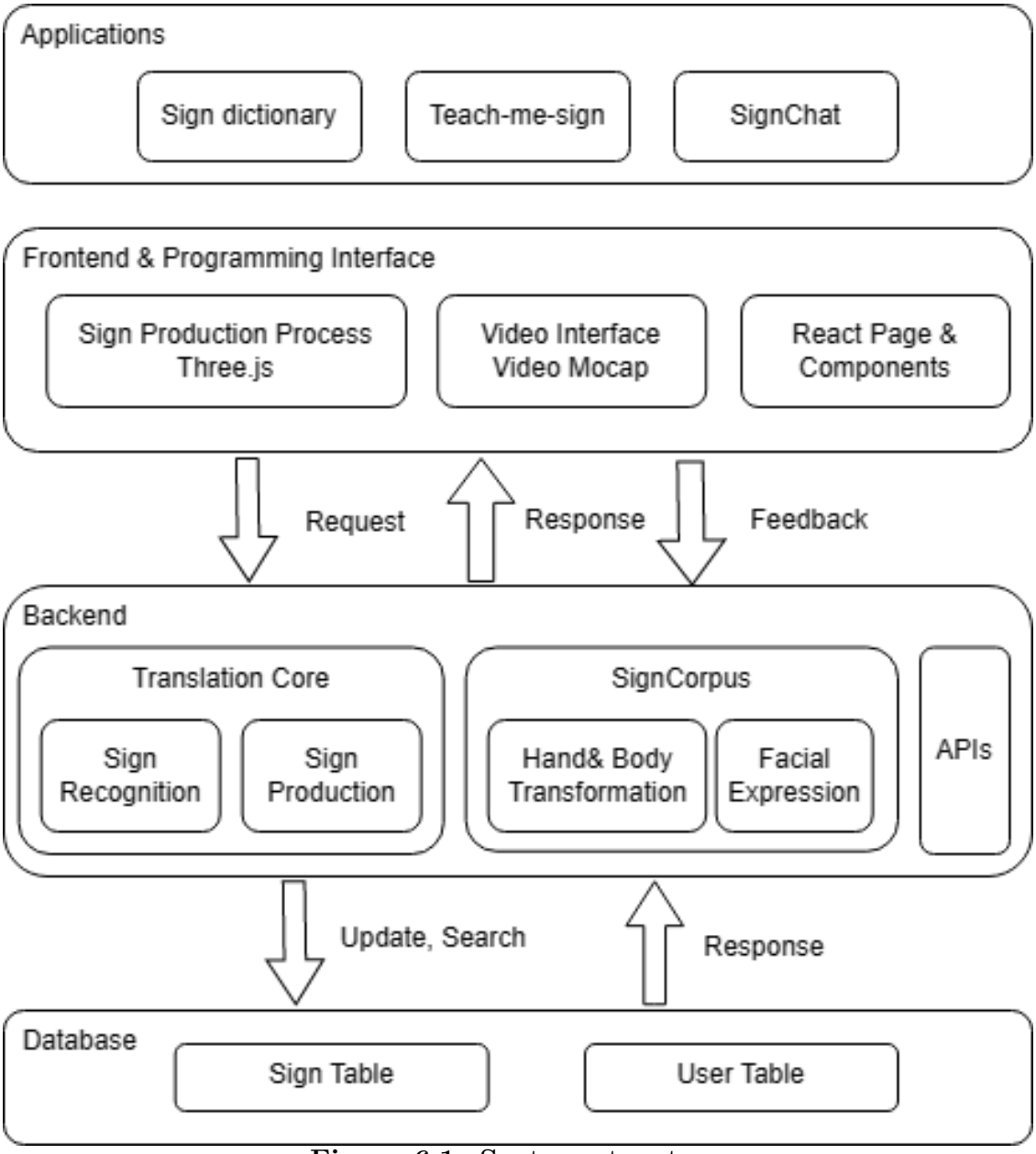

Figure 6.1: System structure

demand. This section is dynamic and can change according to different development need. The footer is attached to the end of each page and serves as a copyright statement for the website.

In summary, the interface layout provides a clear and intuitive navigation structure that makes it easy for users to find what they are looking for. Each component serving a distinct purpose to enhance the overall user experience.

6.1.2 Backend

The backend is an essential component of the system, facilitating seamless communication between different parts and enabling smooth data processing. We selected Flask as the backend framework for its compatibility and ease of transition from the existing legacy system. The backend is accountable for connecting to the database and accessing the algorithms and models to handle incoming data. The detailed design of each module will be discussed in the following sections.

Moreover, the privacy and security of the confidential information stored in the database are of utmost importance. To ensure this, the port used for the backend has been kept private and set to 6000, inaccessible from the public internet. This helps to thwart unauthorized access and preserve the confidentiality of the data being processed by the backend.

6.1.3 Database

The database *AnySign_web*uses MySQL as the management system. We designed several tables to serve different use cases. One table contains all words with the necessary information, including their ids, parameters, landmark file location, etc. The other table is used for user system with the user's email, area, and identity information. The datasets introduced below is where our raw data comes from.

The database *AnySign_web* utilizes MySQL as its database management system and

has been designed with multiple tables to cater to different purposes. One of the tables holds a comprehensive list of all words with relevant information such as their unique IDs, parameters, and landmark file locations. Another table is dedicated to the user system and contains user-specific information such as email, location, and identity.

Video dataset We designed a motion-capture (MoCap) [RKE21a] based approach to produce each sign with animated avatars. The motion landmarks for the avatar were sourced from the WLASL video dataset, which is the largest video dataset for ASL recognition and features 2,000 common ASL words.

Sign parameter dataset The sign parameter dataset is based on the ASL-LEX 2.0, which is an open-source database of the lexical and phonological properties of ASL signs. The 2.0 version of the database has been significantly expanded and now includes information on 2,723 signs. It provides a comprehensive introduction to each sign, including handshape and sign type, which is useful for our sign search module.

6.2 Sign Dictioanry

The sign dictionary system aims at facilitating easy and intuitive learning about signs. The system has two primary functionalities, search and results display. The user can input the desired sign they wish to search for and receive a list of cards displaying relevant information about each sign, along with its corresponding animation. The information displayed in the results cards is comprehensive, and our motion-capture-based avatar provides a vivid and interactive representation of each sign, making it easier for users to understand and learn.

This system serves as a one-stop-shop for users seeking information about ASL signs and makes it easy for them to access and learn about each sign. The user-friendly interface and intuitive design of the search engine make it a valuable tool for anyone looking to learn about ASL signs, whether they are a beginner or an experienced user.

```
{
    "pose_landmarks": [
    {
        "x": 0.5,
        "y": 0.5,
        "z": 0.0,
        "visibility":0.99
    }, ...] (33 landmarks in each frame),
    [...] (landmarks for rest frames)
  ],
    "face_landmarks": [...],
    "left_hand_landmarks": [...],
    "right_hand_landmarks": [...]
}
```

Listing 6.1: Landmark JSON file structure obtained from video.

6.2.1 Animation processing

As introduced in the 6.1.3, the motion landmarks are obtained from the WLASL video dataset. There are 10908 json files in total, each containing *pose_landmarks*, *face_landmarks*, *left_hand_landmarks*, and *right_hand_landmarks* in chronological order per frame. The landmarks follow the MediaPipe[b820] pose 33 3D landmarks structure (shown in 6.1), which is an ML solution for body pose tracking from video frames. However, our 3D avatar model requires a dictionary input format with keys such as "Spine" and "Head" for the body, and a 52-dimension facial morph dictionary (ARKit)[b9]. To process the landmarks accordingly, we developed *SignCorpus*, which incorporates modules for generating and refining sign corpus, including facial expressions and *ik.py* for hand and body transformation. We used *inverse kinematics(IK)* to get the body and hand animation data *morph_target* and then used facial expression module to produce the *morph_target*.

```
{
    "1": [
        "trans_dict": {
            "Spine1": [...],
            ...
        },
        "morph_target": {
            "browDownLeft": 0,
            "browDownRight": 1,
            ...
        }], (the first frame)
    "2": [...],
    ...(rest frames)
}
```

Listing 6.2: Animation JSON file structure after processing.

6.2.2 Database

Each word in our database is assigned a unique ID (as referenced in 6.1.3). However, it is possible for a single word to have multiple glosses, meaning that there can be multiple results of one sign. Additionally, the quality of the videos associated with each gloss can vary. In our initial version, we display only one animation for each word from the WLASL dataset. To determine which animation to display, we mapped each video file size to the corresponding sign and selected the animation with the highest quality. Thus, we can only determine which sign animation has the largest video file size for each gloss in the raw data.

6.2.3 User Interface

The interface of the sign dictionary system consists of two main pages: a search page for signs and a results page displaying sign cards with animations.

Sign search page The sign search page, which also serves as the home page of the website, has a simple and user-friendly interface, as depicted in Figure 6.2. The center of the page features a search text input field. Users can initiate a search by clicking the search

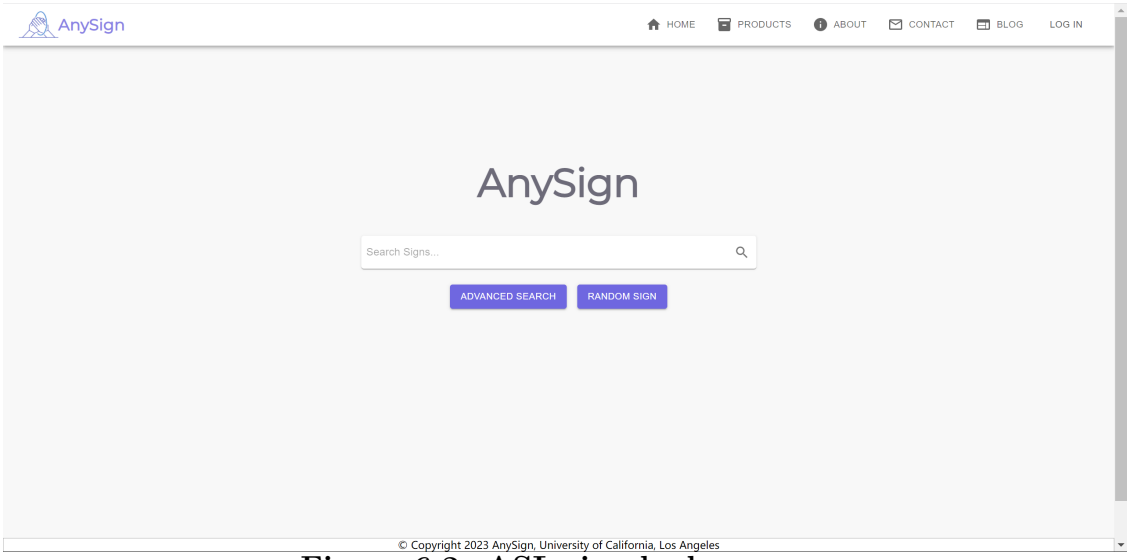

Figure 6.2: ASL sign lookup page

icon or pressing the enter key, which will redirect them to the results page. The sign input is automatically included in the URL, enabling users to easily visit specific pages and track their activities through the URL.

Search result page In the results page, all signs that match the input will be displayed as a list of cards, as shown in Figure 6.3. Each card features an avatar canvas on the left and sign word, description, and a play button for the sign animation on the right. When the play button is pressed, the static avatar picture will be replaced with an ASL video demonstrating the gestures of the sign, as depicted in Figure 6.4.

Upon being redirected to the results page, the React component retrieves the user input from the URL and sends a request to the backend. Upon receiving the result data, each matching sign information is mapped to a *SignCard* component. The *SignCard* component is responsible for the card layout and managing video playback. If the user presses the play button, the *SignCard* component sends a request for animation data and passes it to the *SignAnimation* module, which is responsible for the avatar actions.

The *SignAnimation* module uses Three.js to load and display animated 3D avatars in a web browser. It exports an *animationDrawer* function, which can load 3D models saved

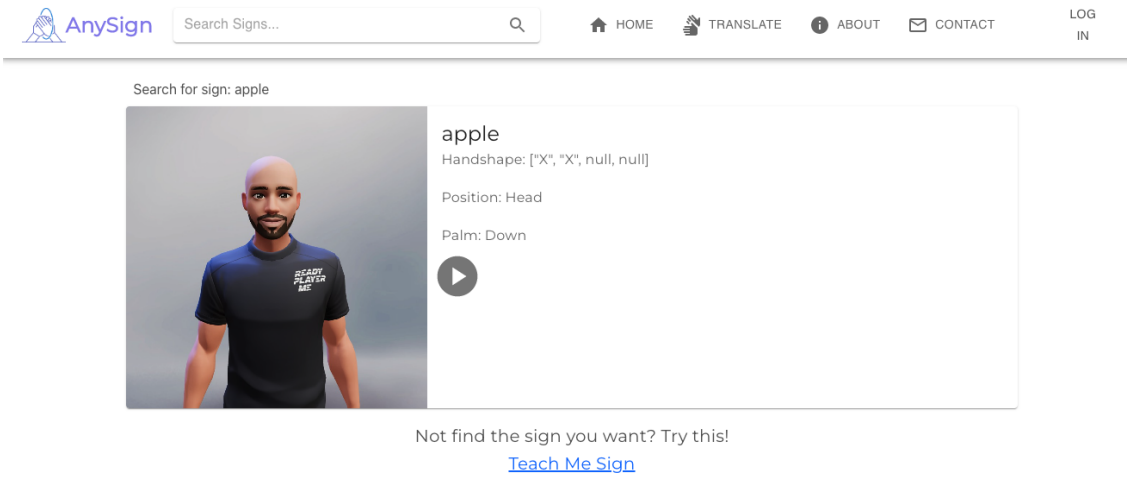

Figure 6.3: Result page for each sign.

in the GL Transmission Format (glTF) and set the model gestures based on the animation data previously obtained. Each animation plays in a loop unless the user presses the pause button or another card's animation is played. Only one animation is designed to play at a time, which is achieved through the use of the React embedded component, Reducer. The Reducer consolidates all state update logic outside of the component into a single function. The result cards are wrapped in React context to maintain a shared status *glossID*. The Reducer updates the *glossID* status of the currently playing video whenever the play button is pressed. Each sign card has a unique ID stored in the database, so when the currently played video ID does not match the one in the Reducer, the animation will automatically pause.

6.2.4 APIs

The search sign module has two APIs available. The first API is responsible for returning all matched signs from the database, while the second API is responsible for returning the corresponding animation data for a given sign.

For the search API, the module retrieves the user input from the request and fetches all

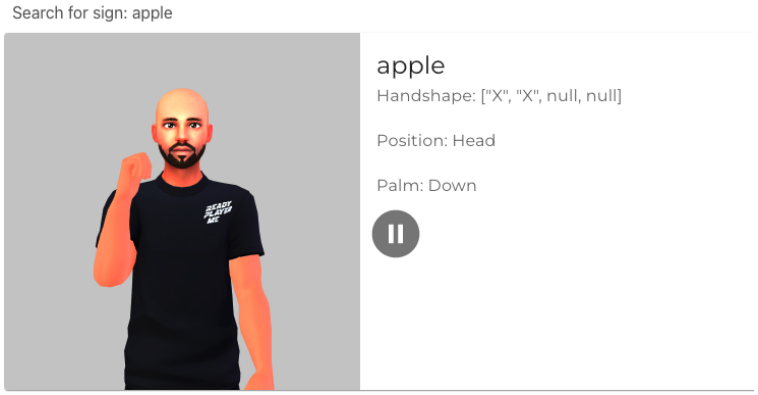

Figure 6.4: Card interface when playing animation.

data that match the input from the database. The module then selects the result with the largest video size and returns the associated ID, gloss, and related explanation.

On the other hand, the sign/animation API is activated when the user presses the play button on the frontend. This API sends a request with an ID, which the backend uses to locate the corresponding animation data and return it to the frontend.

6.2.5 Let ML Model Learn the Signs from the Dictionary

A significant challenge faced in the field of sign language recognition stems from the scarcity of video data for training machine learning models. The relatively limited number of samples available for ASL, as compared to major spoken languages, compounds this issue. Nevertheless, an underexplored approach that can potentially mitigate this problem involves leveraging the ASL sign dictionary as a training source for the recognition model.

The fundamental idea behind this strategy involves exploiting the inherent structural composition of sign language, specifically the discrete features that form the basis of signs. Each sign can be decomposed into a combination of several discrete features, including handshape, hand symmetry, wrist position, orientation, movement, and more. These features can be captured in the ASL dictionary and subsequently reorganized for machine processing.

In the proposed approach, the first step involves the extraction of hand landmarks. These

landmarks encompass crucial information, such as the handshapes and their symmetry. The second step considers the wrist position and orientation, which are used to identify hand number, movement, and palm orientation. These landmark traces then feed into a classification model, which outputs discrete classifications.

Next, we combine the discrete classifications to identify potential signs that match the same category. Given that the signs within each category are sparse (with 74.7% of the signs being unique within their respective categories), this step considerably narrows down the possibilities.

However, the categorization process may introduce errors, owing to the inevitable overlap and ambiguity in sign features. Subsequently, the signs can be classified based on similarity scores, calculated as the cosine distance between the feature vectors and the ground truth. In this way, even with limited video data, the model can effectively learn and recognize ASL signs by referring to the sign dictionary.

6.3 Teach-Me-Sign

ASL is a language with a wide range of regional variations, as different areas and people may have distinct body shapes and habits for producing the various handshapes used in the language. To ensure that our system accurately reflects these regional variations and diverse expressions, we have developed the teach-me-sign system.

The teach-me-sign system is designed to help us collect a comprehensive sign corpus that accurately represents the full range of regional ASL variations. To achieve this goal, we utilize a motion capture (MoCap)-based approach, which will be introduced in the following sections.

Users can access the "Teach-Me-Sign" system in two ways: either through the "Teach-Me-Sign" link in the sign dictionary results page or by using the random sign button on the search page. There are several use cases for the system. The first scenario is when a user

finds a sign animation in the database, but notices variations in how the sign is displayed. Alternatively, there may be signs that are not included in our corpus, prompting the user to teach our avatar their way of performing the sign using the "Teach-Me-Sign" link. In the second scenario, users can access the system by teaching the avatar random signs. This can be especially useful when users do not have a specific sign in mind that they wish to teach.

Our system plan to prioritize signs that are not well-represented in our database if users choose random signs, and we plan to generate a ranking list of users based on their contributions to encourage them to add more sign data. Currently, signs are randomly selected with equal probability, and users can input any sign they wish to teach the avatar.

As for the specific functionalities, users are able to record their gestures for signs. The avatar will represent the animation that has been recorded for the user to verify the animation meet their expectation. Once they are satisfied with it, they can submit it to the server. A user system is built to help collect the area and identity information to help further process the data received.

The system allows users to record their own gestures for ASL signs. This feature is designed to enhance the inclusivity of our system, by allowing users to contribute their own unique regional and personal variations to our sign corpus.

When a user records their gesture, the avatar will display the corresponding animation, allowing the user to verify that the avatar correctly interprets the sign. Once the user is satisfied with the animation, they can submit it to the server.

To help us better process the data we receive, we have built a user system that collects information about the user's region and identity. This information can help us on the further learning and processing.

6.3.1 User Interface

"Teach-Me-Sign" has a single user interface page as depicted in Figure 6.5, which is primarily composed of two canvases. The left canvas shows an avatar, while the right canvas displays the user's video stream. The avatar is designed to follow the user's movements when they are initially setting up or recording a sign. This allows users to see how their movements correspond to the animation in real-time. In addition, the video canvas displays the landmarks that have been recognized by our model. This feature allows users to see how our avatar is interpreting their movements and provides valuable feedback that can help users refine their gestures and improve the quality of submitted data.

At the top of the page, there is a user input area where the user can enter the sign they wish to teach. This area also includes a button that allows users to either return to the result page or choose a random sign, depending on how they accessed the page. The control panel for the video source is located in the lower-left corner of the page. Here, users can monitor the fps and select the device they wish to use for their video input. The record and submit buttons are located in the lower-right corner of the page. When the user is ready to record their sign, they can press the record button, and the avatar will repeat the sign after it has been recorded. If the user is satisfied with the result, they can press the submit button. Once the backend successfully saves the animation, a modal will appear with a thank you notice.

6.3.2 Animation Module

The animation module is a key component of the teach-me-sign. It relies on two custom components - *MoCap.js* and *avatar.js* - to manage video input and avatar output separately while communicating with each other and the backend.

The *MoCap* component plays a crucial role in capturing each video frame and employing MediaPipe Holistic to produce landmarks. MediaPipe Holistic amalgamates pose, face,

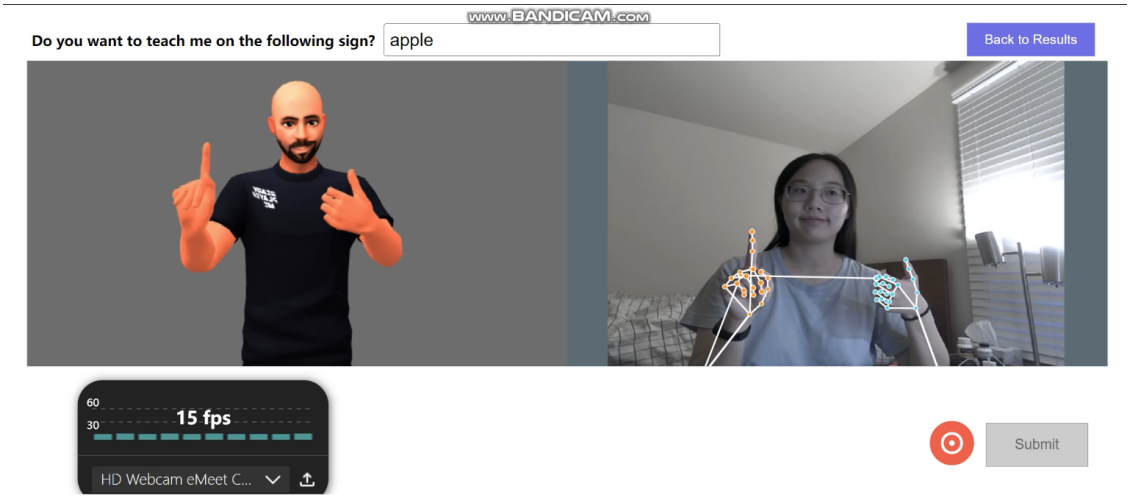

Figure 6.5: Teach-Me-Sign Interface

and hand landmark models, collectively generating a comprehensive 543 landmarks. These encompass 33 pose landmarks, 468 face landmarks, and 42 hand landmarks (21 per hand).

However, as discussed in section 6.2.1, the *avatar* library cannot directly consume these landmarks. Therefore, they undergo processing by the backend before being transmitted back to the *avatar* library. Furnished with the corresponding skeleton data, the *avatar* component can then render diverse animations that accurately mirror the ASL signs that users have either recorded or generated.

This system is further enhanced by the inclusion of animations from WLASL, deployed using a combination of skeleton and morph animations. Such a setup ensures the highest level of accuracy and detail in representing ASL signs in an animated form.

Facial Expression Module While our avatar is capable of displaying hand and body movements, facial expressions are also an important aspect of ASL that are not currently captured by the system. To address this limitation, we plan to add a facial expression module to our *SignCorpus* library.

The facial animation in our system is based on shape keys, and we have updated our avatar from the ready-player-me Metaverse 3D Avatar Creator[b11] to include shape keys

for facial animation control. To create the facial expression module, we designed the module based on an open-source package called FaceTranslator[M]. This module converts MediaPipe face motion to shape keys by projecting the landmarks onto the canvas and using the *BlendshapeCalculator* component to calculate the corresponding morph representations. The output of this process is an ARKit morph dictionary, as introduced previously.

Due to limitations in the accuracy of facial landmark recognition, we have chosen to selectively display facial keys for certain facial features, such as eyebrows and mouth. For features that are not displayed, the corresponding keys in the morph dictionary are set to zero in order to maintain the dimensionality of 52. By incorporating facial expressions into our system, we hope to create a more comprehensive and expressive representation of ASL.

6.3.3 APIs

We rely on two APIs to support the teach-me-sign system. The first API is used to facilitate the process of transforming video landmarks into motion animations. The client sends the video landmarks to the server through this API, and the server is responsible for processing the data and generating the corresponding motion animation.

The second API is used to save the animation data after the user submits their recording. This allows us to efficiently store and organize the new sign data that users have contributed.

6.3.4 User System

A distinctive challenge in effectively harnessing ASL within a digital platform lies in the inherent variability of the language. As noted earlier, ASL usage can differ markedly across regions, communities, and individuals. This variability adds a layer of complexity to the accurate interpretation and representation of ASL.

In light of this, we underscore the necessity of developing a robust user system on our platform. This system is not simply an add-on but a critical component of our strategy to

enhance the accuracy and usability of our platform. It is meticulously designed to capture and learn from these regional and personal nuances in ASL usage.

By integrating this user system into our website, we aim to foster a more comprehensive understanding of these differences, thereby facilitating a more accurate translation and representation of ASL. This system not only aids us in comprehending the idiosyncrasies of individual users but also allows for the improvement of our platform's overall effectiveness and adaptability, making it a truly user-centric resource for the ASL community.

Located in the navigation bar, the login button steers users toward the login page. As it stands, our system supports login authentication via Google email accounts. Upon successful account authorization, users gain access to their personal profile page (refer to Figure6.6).

Within this personalized space, users possess the ability to modify their name and contribute details regarding their geographical region and linguistic background. This granular level of information, coupled with the animation data submitted by users, plays a vital role in enriching our understanding of the users' contexts. Such understanding equips us to provide ASL services that are not only accurate but are also tailored to the unique needs and backgrounds of individual users, thus enhancing the overall user experience on our platform.

6.4 English to ASL Translator

The translator module is primarily based on the original module and is designed to allow users to input English text and view the output in ASL grammar, as well as an animation demonstrating how to represent it.

The feasibility of implementing an ASL translator is due to one unique feature of the language: the limited number of handshapes. ASL is capable of representing a wide range of words and concepts. In some cases where no corresponding sign exists, ASL can use fingerspelling to manually spell out a word letter by letter using hand gestures that correspond to each letter in the English alphabet. This feature enables our translator to display all words

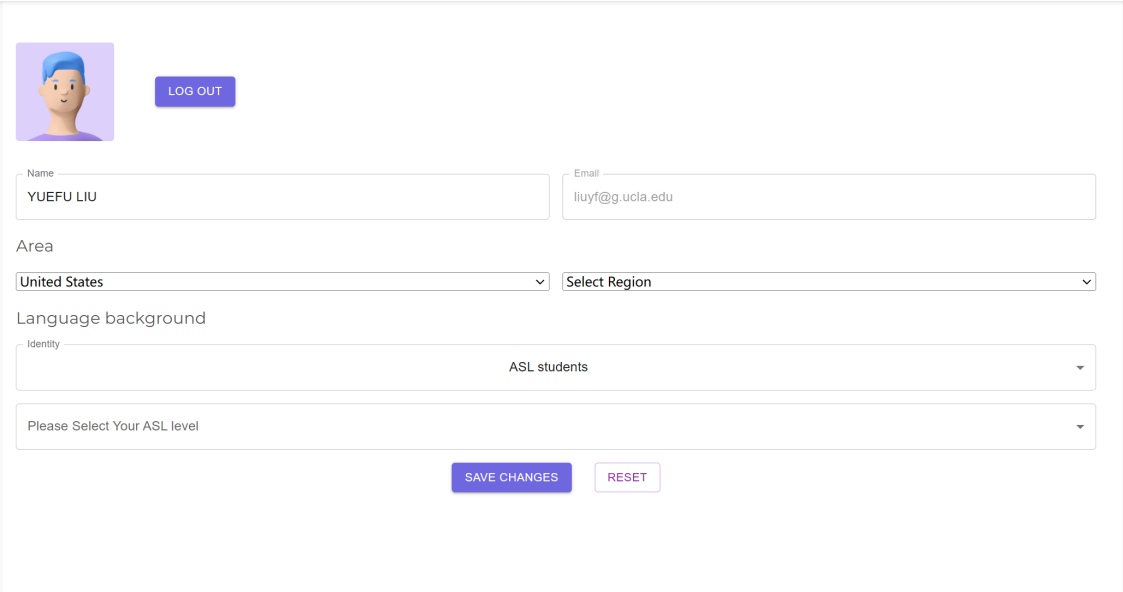

Figure 6.6: Profile page for identifying user's sign language background

in English, including those that do not have a corresponding sign.

6.4.1 User Interface

The central focus of our ASL translation system is the translator interface (`https://dev.anysign.net/e2a`), an interactive application designed for the facilitation of ASL learning and practice. The interface, comprised of several key components, caters to users' needs by displaying interactive translations of ASL sentences in an easily understandable manner.

Canvas and Animated Demonstrations The canvas, situated in the center of the screen, serves as a dynamic platform for visualizing translated ASL sentences. The default animation sequence starts with the sign "HELLO". The word or letter that is currently being signed is displayed at the lower-right corner of the canvas, providing an instantaneous reference for users. This visual aid contributes significantly to enhancing users' comprehension of ASL.

English Text Input and ASL Translation Display Located directly beneath the canvas on the left-hand side of the screen is a text field for user input. Here, users can enter

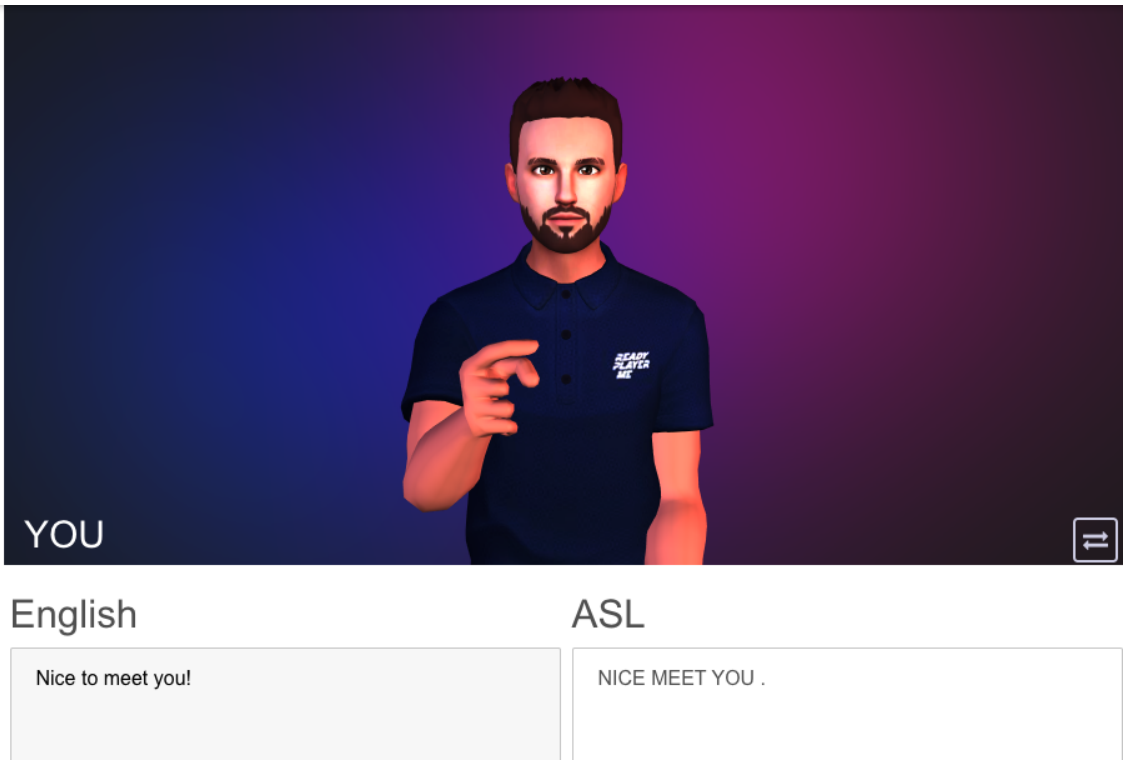

Figure 6.7: English to ASL translator

the English text they wish to translate into ASL. Upon clicking the translate button, two simultaneous actions are triggered.

First, the ASL gloss sequence, which provides the structure and order of signs in the translated ASL sentence, appears on the right-hand side of the screen. This feature allows users to understand the grammatical flow of the ASL sentence. Concurrently, the canvas comes to life, showcasing an animated demonstration of the translated ASL sentence.

This dual response system allows users to connect the English text with its corresponding ASL gloss sequence while witnessing a real-time demonstration of the sentence. Through this innovative interface, the system aims to enhance the user experience by making ASL learning more engaging, effective, and intuitive.

6.4.2 APIs

The translator module is powered by a single API that is responsible for converting English text into ASL representations and associated animation data. This API is accessed using the endpoint `e2a_animation/<english>`, where the input is an English string and the output is the corresponding ASL sentence and animation data.

Once the user inputs English text and initiates the translation process, the frontend sends a request to the e2a_animation API. The API utilizes the algorithms and tools provided by our custom package, *translation-core*, to process the input. This package provides interfaces for both English-to-ASL and ASL-to-English translation.

Once the translation process is complete, the frontend updates the text field to display the ASL output. At the same time, the animation data is loaded onto an avatar on the screen, providing a visual representation of the ASL sentence.

6.5 SignChat

SignChat is an innovative system that aims to bridge the communication gap between chatbots and sign language users. Our solution has been developed to enable sign language interfaces to advanced chatbots. With SignChat, signers can interact with chatbots and receive responses in their preferred language.

To ensure accurate sign language interpretation and production, we have integrated Large Language Models(LLMs) into our solution. We also apply them to understand and generate natural sign language, providing a more authentic communication experience for signers.

Moreover, SignChat is designed to learn from signers' feedback, enabling it to handle variations in signs and grammar and continually improve its performance over time.

We have chosen to implement our solution to work with ASL as it is one of the most widely recognized and used sign languages globally. Our system is scalable and can be

adapted to other sign languages as well.

6.5.1 Pipeline

Our goal is to develop a chatbot model capable of understanding, producing, displaying professionalism and learning variations in ASL. To achieve this, we have developed a three-step solution to decompose the task. We uses a dialogue core consists of sign recognition, discussion, and production modules to implement the solution.

Interpretation from ASL. Pre-trained large language models are designed to take text as input and cannot directly accept input from ASL gestures. Therefore, the first step is to convert ASL gestures into English text, which can then be processed by the language model as machine-readable input.

ASL has its own vocabulary and grammar rules that are differ from English. To accurately convert ASL into English, we follow a two-step approach of recognizing glosses (a written word that represents the meaning of a sign) and translating gloss sequences into English sentences. Accurate recognition and high-quality translation in traditional approaches require large training sets as input. However, ASL does not have a well-accepted written form and is instead typically recorded using video recordings. This lack of documentation makes it difficult to obtain high-quality interpretation from ASL.

To overcome this issue, we propose the following steps to design an effective solution: 1) use a n-gram model to help select the correct word from the candidates, and 2) apply grammar correction and improvement to generate English from the ASL expression.

Our approach to achieving our goal involves utilizing pre-trained LLMs, specifically a 2-gram model and GPT-3 [BMR20]. The 2-gram model is trained on both English and ASL corpus, which assists in selecting the most appropriate sign by considering the context and language usage.

When the server receives landmarks from the sign capture interface, our sign translation

module produces a set of potential sign selections. We then apply the previously mentioned 2-gram model to these options, select the most appropriate sign based on context, and use it to generate the gloss sequence.

Additionally, we utilize GPT-3 for grammar correction and in-context learning to further refine the translation. Specifically, we fine-tune GPT-3 to improve the accuracy and fluency of the translation when converting the ASL gloss sequence to English. This improves the quality and accuracy of the final translation output.

ASL Production from Response. Producing sign language from English language models is challenging for long sentences since English and ASL have different grammar rules. When a user queries the chatbot, the responses are provided in English text. However, ASL signers may still prefer to receive the response message in sign language due to their language background or learning purpose. To meet this need, we leverage Large Language Models (LLMs) for pre-processing our English input and then use the processed input to generate an ASL response.

First, we use GPT-3 to generate a response to the English input obtained from the sign recognition, with a prompt like "The user input is: [English sentence]. Your response is: ". Next, to balance response speed and quality, we utilize a large-sized T5 model to extract the topic of sentence sequences. We then use a pre-trained GPT-3 model to convert the long response to short sentences with simple grammar structures, making it easier to transfer English to ASL. This step includes a prompt like "Please refine and shorten 'Please make these sentences shorter'". Afterward, our dialogue core model converts the English to a gloss sequence, and finally, we produce the final sign gloss sequence using a hard-coded ASL sequence generation.

Learn the Sign Variations with Human Feedback. As previously mentioned, ASL includes a significant number of variations that are difficult to document. Existing corpus data is inadequate to train a model that can convert between English and ASL. To overcome

this difficulty, our system learns from signers on two key levels. First, the system learns to recognize variations in signs used by different signers. If one detects an inaccurately recognized gloss, the user can correct it in the input text area. The system records this information and uses it to improve the gloss corpus, which forms the basis for sign recognition. Second, our system learns to produce more natural expressions from user feedback on word order. Both the English response and the ASL gloss sequence are provided in the output text area, and users can edit the gloss sequence. These refined gloss sequences are used to fine-tune the topic-comment model for sentence generation. Through this iterative process of collecting feedback and refining the models, our system can continually improve itself over time.

6.5.2 User Interface

The web page for the signchat system is similar to the one used in teach-me-sign and translator. There are two interfaces respectively responsible for sign capture and sign production.

For sign capture, we first apply a skeleton-based approach for streaming and recognizing signs from the webcam video. We deploy a pose recognition model, Mediapipe, on the web page so that the browser detects and skeleton landmark sequences. The detected landmarks contain the major positions of the body, hands, and face, which is the same as in Section6.3. These landmarks are processed in on the server using *translation-core* package.

After processing the landmarks following the steps in our pipeline, the ASL Sign production interface then displays the ASL animations. the ASL responses from the chatbot are streamed back to the server as animation sequences. These animations are then dynamically applied to the pre-loaded avatars on the client's browser for real-time display, without the need for any pre-rendering on the server. This results in an engaging and interactive experience for the users.

While our system can provide ASL-to-ASL chat functionality by combining the two inter-

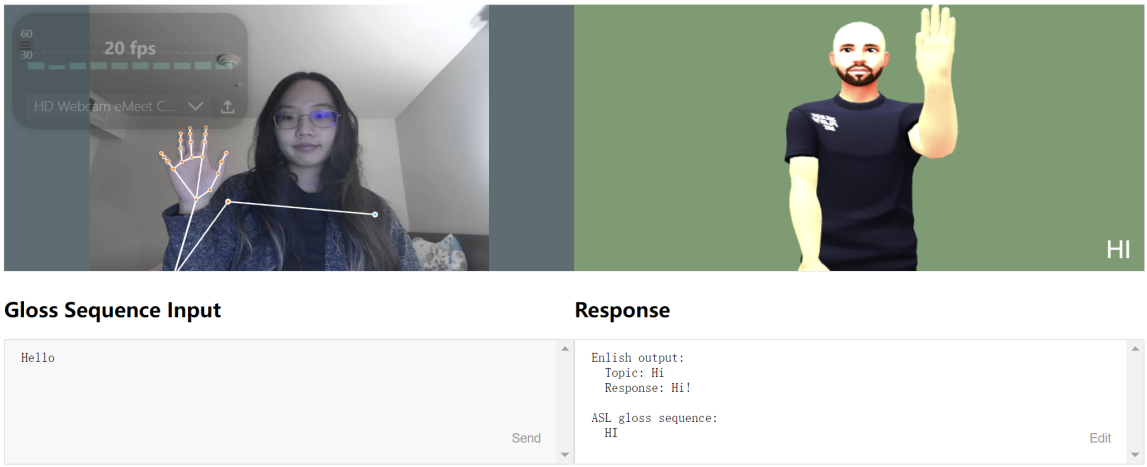

Figure 6.8: SignChat Interface with Avatar Response.

faces, we recognize that accurately converting between English and ASL can be a challenge. To overcome this challenge, we have incorporated a feature that allows users to contribute feedback to improve the system's accuracy.

To facilitate this, our interface provides two text areas that display the gloss sequence. When a user inputs gestures, the corresponding ASL gloss is displayed in the gloss sequence input area, with the response sequence appearing in the response area. Users can then review the gloss sequence and animation, and make any necessary edits to provide feedback on any inaccuracies.

6.5.3 Downstream Tasks

A sign language interaction platform has the potential to revolutionize the way people interact with and learn sign languages. It can be used in a number of downstream tasks to improve accessibility for sign language users. The following are some of the key downstream tasks for a sign language chatbot:

Enhancing Sign Language Learning through Interaction. Interaction is a crucial component of language learning. With the emergence of sign language chatbots, students

can now practice their sign language skills and receive real-time feedback from virtual tutors. This immersive learning experience can help bridge the gap in sign language education and provide students with a valuable tool to improve their signing proficiency.

Inclusive Educational Tools and Resources. Education is a fundamental field where sign language chatbots could make a significant impact. Often, educational resources and tools are not designed with accessibility for sign language users in mind. Integrating a sign language chatbot into educational platforms could create an inclusive learning environment. For instance, it could enable real-time translation of lectures and learning materials into sign language, assist in virtual classroom interactions, and even help design curricula specifically for sign language learners. Furthermore, coupling such a chatbot with Assistive AR could lead to immersive, interactive learning experiences that cater to the needs of both sign language users and those wishing to learn sign language, fostering an inclusive educational landscape.

Improving Accessibility for Sign Language Users in Customer Services. Customer services can be challenging for sign language users to access, particularly if they are not designed with accessibility in mind. Sign language chatbots can serve as accessibility interfaces by translating text and voice into sign language and vice versa. This approach can help ensure sign language users can fully participate in online services and access the same information and opportunities as all other users.

Facilitating Virtual Assistants and Assistive AR Integration for Sign Language Users. While voice-activated virtual assistants like Siri have become commonplace, there's a dearth of virtual assistants designed specifically for sign language users. SignChat, integrated with modern hardware such as smart monitors, smartphones, and Assistive AR goggles, has the potential to fill this void. It could serve as a virtual assistant for signers, enabling applications currently available for voice assistants - including home automation and entertainment - to be accessed by sign language users in an intuitive and accessible way.

Furthermore, the integration of Assistive AR technology could offer real-time sign language translation, converting signed phrases into text or speech and vice versa. This innovative combination could serve as a communication bridge in various settings, including education, healthcare, and professional environments, truly enhancing inclusivity and communication.

The downstream tasks of a sign language chatbot have the potential to greatly improve accessibility and life quality for sign language users and provide new opportunities for language learning and interaction.

6.6 Future Work

There are several directions for future research on AnySign platform. One direction is to expand the functionality of the Sign Dictionary module to allow for multiple kinds of signs to be displayed. To achieve this, we plan to cluster the sign data in our database and incorporate users' submissions to update the original database. With this effort, signs from various regions and some newly accepted signs can also be displayed, making the platform more comprehensive.

Another area of focus for future research is the backend functions of the entire system. Although the dialogue core package and sign corpus processing and generation modules have been developed, some of the backend functions still rely on local data corpus. In order to improve the efficiency of the system, we plan to directly import the package and connect all APIs to the server database for loading and updating.

6.7 Conclusion

In conclusion, AnySign is a comprehensive cross-device platform that aims to make ASL more accessible. Our approach addresses the challenges faced by the deaf community, providing a collaborative solution to improve ASL communication. The Sign Dictionary, English to ASL

Translator, and Teach-Me-Sign modules work together to create a comprehensive corpus of ASL. Our solution, SignChat, is a novel chatbot that bridges ASL with state-of-the-art AI tools, bringing the benefits of AI advancement to the sign language community. Our system has received positive feedback and demonstrates the potential for further advancements in sign language chatbots and their applications in various fields. AnySign is a promising solution to the challenges faced by ASL human-machine communication and can demonstrate the potential for further advancements in various fields.

CHAPTER 7

MORSE: Private Model Protection for Wearable and IoT System

While the previous chapters focused on delivering ASL interaction with mobile and wearable devices, such as assistive glasses, user privacy remains paramount—especially when sensors gather personal data. This chapter presents a protection scheme designed to safeguard ML models that contain private or personal information.

The Internet of Things (IoT) applications are envisioned to enable unprecedented cyber-physical interactions in an automated fashion. In a typical IoT usage scenario, multiple sensor data streams form the system inputs. An IoT service provider is interested in learning their relationship with the system output. To this end, the input-output data is fed to the IoT service provider, which may further train a model using machine learning or other models at an edge or cloud server. Once trained, the model can be used to estimate or predict future output given certain inputs.

The above model-centric operations hold great promises for many emerging IoT applications [WJC19, SHB17]. A typical application is shown in Fig. 7.1. The IoT service provider configures sensors to collect patients' health data streams. The inputs are forwarded to the edge server, which trains a model together with the patient's health status as output. With the trained model, the service provider can infer a patient's health status based on his/her health data. Meanwhile, it wants to protect the model training process from the adversary. We will use the notions of "sensors," "edge," and "service provider" in this chapter.

To protect the model from the adversary over-the-air and the untrusted edge server, the state-of-the-art approach is to encrypt the data during the process of data transfer and processing. In this chapter, we take a different approach. We address a simple yet fundamental question for IoT security: *Can we directly secure the model rather than the data streams prior to training?* We thus explore a paradigm shift from data-centric protection to model-centric security for IoT applications. The perceived benefits to IoT systems are crystal clear. Compared with other alternatives, such as end-to-end encryption, particularly the homomorphic encryption-based scheme [Gen09, SHB17], the processing overhead could be greatly reduced if we only protect the model. So is the power consumption for the energy-constrained IoT devices.

We have two goals for model security design. First, we ensure that an adversary or the untrusted edge server cannot learn the true model with the cleartext input-output data streams. Instead, they can only train an obfuscated model, which does not leak any critical information on the model. Second, the authentic service provider can reconstruct the true model from the obfuscated one. Therefore, we devise a single solution that achieves these two seemingly conflicting goals.

Our study yields a positive answer to the above problem. We use the Bayesian network model as a case study. Specifically, we propose a sampling-based scheme to ensure the independence between the inputs and output. Note that sampling is a natural and inherent procedure with sensors in IoT systems. We leverage the built-in sampling function for model obfuscation. Our solution configures the sensors to collect the data with a well-crafted *data-dependent* sampling scheme. Despite its simplicity, our proposal neutralizes the dependency among inputs and outputs that are used to train the IoT model. We illustrate the idea in §7.2.

We thus design MORSE (**M**odel **O**bfuscation fo**R** **SE**curity), which offers a model-based protection scheme (§7.3). The core of MORSE is a low-overhead sampler that generalizes our data-dependent sampling idea to obfuscate Bayesian network models. By breaking the

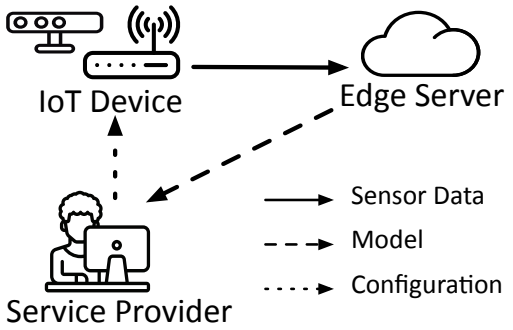

Figure 7.1: Setting for a typical IoT system.

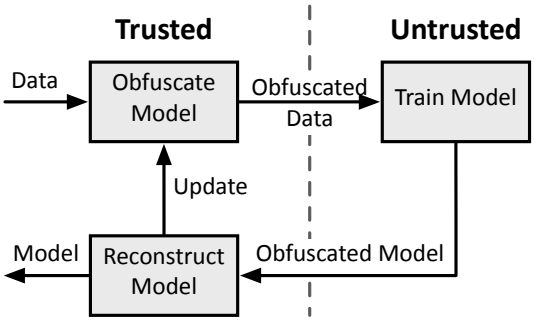

Figure 7.2: Procedures of model obfuscation.

variables into groups according to their dependencies, the sampler allows the sensors to produce data samples based on the subsets of variables. This component addresses the curse of dimensionality, thus making sampling efficient and practical. Moreover, MORSE designs a shuffling module that processes the data after sampling for certain corner cases and a reconstructor that recovers the true model for the authentic service provider.

In §7.4, we formally prove the security of MORSE against two types of attackers. Without knowing the sampling or shuffling scheme, the adversary cannot learn the authentic model even (s)he is aware that MORSE is used. In contrast, for the authorized service provider, we show that MORSE reconstruction converges to the true model. Therefore, MORSE meets both design goals.

We implement MORSE in our IoT testbed (§7.5) and evaluate the effectiveness with three real-world IoT applications (§7.6). We compare the trained obfuscated model with the true model and an uninformed model that randomly guesses the output. We propose a novel metric $RAUC$ to quantify this. It approaches 1 when the obfuscated model approximates the true model but is close to 0 when it behaves similarly to the uninformed model. We show that an attacker, regardless of knowing the existence of MORSE, can only achieve its $RAUC$ close to 0. On the other hand, the reconstructed models are similar to the true models with $RAUC$s over 0.97. Moreover, MORSE incurs 10X and 15kX less computational overhead compared with AES and Paillier, respectively. It is thus an appropriate solution to resource-constrained IoT systems.

The rest of the chapter is organized as follows. Section 7.1 introduces the system setting and the background. Section 7.2 discusses the threat model and the design goals. Section 7.3 elaborates on the design of MORSE, and Section 7.4 analyzes its security and convergence. Section 7.5 presents the prototype and empirical evaluations. Section 7.6 compares our design with the related work. Section 7.8 concludes the chapter.

7.1 Background for Model Security

7.1.1 Model Security for IoT Applications

We consider a *supervised learning* scenario, where a user wants to train a model based on the collected input-output data stream (e.g., from IoT sensors). The user often delegates the model training to a powerful server with computation and storage capacity (e.g., the edge server). After processing, the edge returns the trained model to the authentic service provider. The edge is not always trustworthy since it is vulnerable to the outsider or insider attacks [WJC19, SHB17].

To secure the model, traditional data-driven approaches protect individual data records by encryption. An adversary, who eavesdrops on the data channel or steals data from the edge, cannot decrypt the data without secure keys. The attacker thus fails to train a model. Subsequently, the edge is also disabled from learning the model. To address the dilemma, homomorphic encryption allows the edge to help train the model without knowing the records [SHD15, SHB17, GDL16, AEH15]. Despite optimizations, the overhead is prohibitive for IoT systems.

Instead of encrypting each data record, our shifted paradigm directly obfuscates the model. By processing the entire data from a global, organized approach, an attacker will fail to learn the correct model even if some individual data records are exposed. This is motivated by the premise that securing the model is less expensive than securing the massive data, thus being a better fit for IoT systems.

Model obfuscation for security: We propose a new concept, called "Model Obfuscation." We present the overall procedure in Fig. 7.2. Concretely, we pre-process the data before learning the model so that the resulting data follows a completely different distribution from the original one. When the data is leaked to the attacker, she can only learn an "obfuscated" model and fail to reconstruct the original model. This obfuscation procedure should thus be mathematically sound so that the attacker cannot learn anything useful to reconstruct the model even with information about the obfuscation method.

Modeling via Bayesian network: Bayesian networks (BN) offer an explainable AI approach and are widely used in IoT and machine learning tasks [DSA11, ZP15, NFO17]. In contrast to black-box modeling (e.g., deep learning or other neural network schemes), they offer a white-box solution to capturing the variables' conditional dependencies. It is thus ideal for taking an occurred event and predicting the likelihood of any known cause.

BN provides a coherent framework to characterize probabilistic relations between variables. For example, given the environmental characteristics, researchers have used it in computing the probabilities of fire [JJ12]. In this work, we focus on parameter learning in BN and design obfuscation schemes to protect conditional probabilities. BN graph structures are known via application domain knowledge in our settings [DSA11].

7.1.2 Threat Model

The adversary is interested in learning the model from the sensor data. To be specific, we target preventing the adversary from learning parameters/probabilities in BN. As shown in §7.1, the model is crucial as it contains the economic value and sensitive information. If the attacker attempts to learn a model other than BN from launching the attack, we will empirically show that our method can still defend against it.

We mainly consider the data leakage at the edge server where we offload the computation. The data stored in edge is a common victim of attacks due to internal attacks (e.g., illegal

company actions [Gua13] or curious employees [Ars15]) and external attacks (e.g., from hackers or competitors [Ver19]). We assume the attacker at the edge is "passive," as it still follows the scheme we configure correctly without actively modifying the data or breaking the system. The assumption is valid as the adversary is more eager to learn the valuable model without being detected instead of preventing us from using the system. In addition, the current high-availability solutions provided by edge use multiple nodes for storage, making it difficult to change them all without being detected. This threat model is sometimes referred to as "curious-but-honest."

We consider two levels of attackers:

- **Basic attacker**: A basic attacker directly trains a BN model from the data leaked from the edge.
- **Advanced attacker**: An advanced attacker realizes the method we use and tries to reconstruct the model.

Assumptions: We make a few assumptions about the threat model. The sensors are on-premise and safe. Communications between all entities are secured during transmission. In addition, the protection of the raw data is orthogonal to our work.

7.2 Sampling for Model Security

We elaborate on the security requirement and motivate the sampling-based methods to obfuscate the model.

7.2.1 Design Goals

Model protection: Given the obfuscated sensor data streams, the adversary cannot learn the authentic parameters in the Bayesian network, regardless of whether she knows the network structure. To ensure the security of Bayesian network, we focus on the *input-output*

independence, i.e., if the observed inputs and output are independent after processing, an attacker surely cannot learn the correct relationship between them. Instead, she will think that the output cannot be classified with the inputs. Presenting such an obfuscated model also prevents the advanced attacker from learning valuable information. For example, if the obfuscated model is the "farthest away" from the original one, the advanced attacker might be able to reverse the procedure and infer the model.

Ability to reconstruct: While the model is obfuscated for the attacker, the IoT service provider must be able to reconstruct the original model. In our context, the parameters for the Bayesian network can be reconstructed with the extra information of how the obfuscation is enforced. The reconstructed model needs to converge to the correct one before the obfuscation. The reconstructing process should be functional for any Bayesian model.

Low overhead: The processing should only incur negligible overhead to even run on low-end IoT devices. In addition, we recall our original purpose of incorporating edge that we wish to offload computation to it. This goal requires our solution to utilize edge server for computation even if the data on it is obfuscated.

7.2.2 Idea: Data-Dependent Sampling

We propose a sampling-based approach to achieve input-output independence. Intuitively, this does not work: sampling is usually employed to keep the distribution unchanged from the original data, yet our target is to obfuscate the model. Although it seems conflicting, we can sample with different probabilities that *depend on the data*: we keep the more frequent input-output pairs with lower probabilities. Henceforth, all the pairs will appear with similar probabilities after sampling. As long as the sampling probabilities are kept secret, the adversary cannot acquire the true model. We call it "data-dependent sampling."

We use a simple example to illustrate the idea. Let A be the reading from a temperature

Temp.	$P_r(A,B)$		$S(A,B)$		$P_s(A,B)$	
	Safe	Fire	Safe	Fire	Safe	Fire
Low	0.40	0.10	0.25	1.00	0.25	0.25
High	0.10	0.40	1.00	0.25	0.25	0.25

Table 7.1: An example of data-dependent sampling.

sensor, with values of "high" or "low." Let B represent the fire event, with values of "safe" or "fire." The system wants to learn the probability $P_r(A, B)$. Suppose the true probability is shown in Table 7.1. We use adaptive sampling probability for different combinations. We denote $S(a, b)$ as the sampling probability when $A = a$ and $B = b$ and $P_s(A, B)$ as the distribution after the sampling.

A sufficient condition is to make P_s equal to 0.25 for any input-output pair. We can achieve so with $S(\text{low}, \text{safe}) = 25\%$ and $S(\text{high}, \text{fire}) = 25\%$. The observed sensor reading and fire event are independent. Meanwhile, the service provider can reconstruct P_r from P_s and S.

7.2.3 Challenge for Data-Dependent Sampling Design

The previous example illustrates the feasibility of model obfuscation using data-dependent sampling. However, the obfuscation scheme should be carefully designed to fit the IoT applications with limited hardware capacity.

The major challenge for the data-dependent scheme comes from the curse of dimensionality. We derive sampling probability for every combination of A and B in our example. Although acceptable for a 2-variable case, the overhead is prohibitive with more variables. It significantly impedes the feasibility since we need to query a huge table of sampling probabilities for each data. Considering the limited storage capacity and computational power on the IoT devices, we need to reduce the sampling complexity without compromising the security goals.

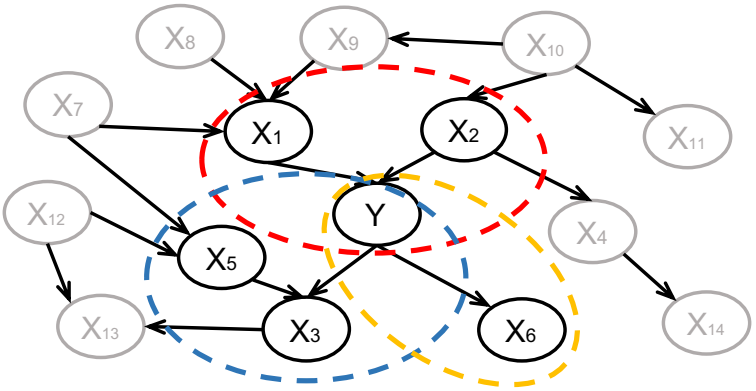

Figure 7.3: A Bayesian network example

7.3 MORSE Design

We design MORSE following the procedures in Fig. 7.2. On the IoT devices, the data are loaded from the source to the obfuscation processor. The processor samples data following the obfuscation scheme and feeds the processed data to the edge server for training an obfuscated model. Finally, the model is reconstructed at the service provider.

This section first elaborates on a data-dependent sampling approach in the Bayesian network, which greatly reduces the overhead. Then we introduce the other components in MORSE for exception handling and online updating. We use the Bayesian network structure in Fig. 7.3 as an example throughout this section.

To facilitate the following component design, we denote the notations we will use in Table 7.2.

7.3.1 Data-Dependent Sampling in Bayesian Network

MORSE scheme generator produces the sampling schemes that eliminate the dependencies between variables. In the example of Fig. 7.3, sampling based on all variables requires sampling probabilities for different combinations of 15 variables. We elaborate on how MORSE reduces this prohibitive overhead by decomposing the scheme using the network structure.

Given the data stream with multiple variables, MORSE applies two approaches to reduce

Variable	Explanation
$\boldsymbol{X}$	n input variables, $\{X_1, \ldots, X_n\}$
Y	Output variable
$\mathrm{MB}(Y)/\mathrm{MB}_s(Y)$	Y's Markov blanket before/after sampling
$\|V\|$	The size of V's alphabet, $V \in \boldsymbol{X} \cup \{Y\}$
P_r	Probability distribution before sampling
P_s	Probability distribution after sampling
P_t	Probability distribution after obfuscation
$\mathrm{pa}(V)$	V's parent nodes, $V \in \boldsymbol{X} \cup \{Y\}$
$S_V(V, \mathrm{pa}(V))$	Sampling probability for $\{V, \mathrm{pa}(V)\}$

Table 7.2: Notations for design and analysis.

the complexity.

1. MORSE splits the variables into groups and applies data-dependent sampling in each group independently. Since the size of each group is small compared to the entire graph, this approach reduces the overhead for storing and querying the sampling probabilities.

2. MORSE only samples according to a subset of groups that include Y to reduce the processing time and complexity.

We now demonstrate how we form the groups and why such a multi-step data-dependent sampling scheme works. We construct groups following the Bayesian network structure. Each group is constructed by a variable V and its parents $\mathrm{pa}(V)$ that it is conditionally dependent on. Note that one variable can belong to multiple groups. We select the groups with Y to process. For our example in Fig. 7.3, we have 3 groups to apply the sampling schemes: $\{Y, \mathrm{pa}(Y)\}$, $\{X_3, \mathrm{pa}(X_3)\}$, and $\{X_6, \mathrm{pa}(X_6)\}$. In the example, the sampling overhead is $\sum_{V \in \{Y, X_3, X_6\}} \left(\prod_{V_o \in \{V\} \cup \mathrm{pa}(V)} b_{V_o} \right)$ which is much smaller compared to $\prod_{V \in \boldsymbol{X} \cup \{Y\}} \|V\|$.

For groups with Y, we use sampling to make $\mathrm{pa}(V)$ independent of V. We use $S_V(V, \mathrm{pa}(V))$

to define the sampling probability given the value of V and $\text{pa}(V)$. When the values of the variables in the group are $\text{pa}(V) = \boldsymbol{v}_{pa}$ and $V = v$, we sample this data with probability

$$S_V(v, \boldsymbol{v}_{pa}) = \frac{1/\|V\|}{P_r(v \mid \boldsymbol{v}_{pa})}, \tag{7.1}$$

where P_r is the probability before sampling and $\|V\|$ is the number of possible values that V can take, i.e., V's alphabet size. Any possible value for V thus appears with equal probability after sampling. After processing all groups, we keep the data if sampled in every step.

Note that the calculated sampling probabilities can be greater than 1 to balance the data size before and after sampling. We randomly select the sampling times from a distribution with the expected value equals to the S_V. If a record is sampled more than one time, the duplicates are inserted after a random delay so that the attacker cannot filter out the repeated samples.

The philosophy of the design is to eliminate the dependency between Y and its close neighbors, which implicitly reduces the dependency between Y and the other variables. We later prove the effectiveness of this approach in Theorem 7.4.1, §7.4: sampling eliminates the dependency within the group without adding any new dependency. It implies that, by applying the sampling scheme to the groups with Y, our protection goal of input-output independence is reached.

7.3.2 Other Components of MORSE

Based on the data-dependent sampling introduced above, we further enhance MORSE with exception handling and an online updating scheme as the general solution.

Obfuscation for 0-probabilities: Sampling scheme in Eq. (7.1) is viable when $P_r\left(v \mid \boldsymbol{v}_{pa}\right)$ is greater than 0. When it equals 0, one common solution is to assign a small value to the entry using Laplace smoothing [AC98]. We show a more rigorous approach that keeps input-output independence without introducing much computation complexity and storage

cost.

We select the nodes form groups with 0-probability entries and denote this set of nodes except Y as $\mathrm{MB}_s(Y)$, the Markov blanket [Pea14] of Y after sampling. In Fig. 7.3, suppose group $\{X_3, \mathrm{pa}(X_3)\}$ has a 0-probability. Then, $\mathrm{MB}_s(Y) = \{X_5, X_3\}$. The post-sampling distribution is

$$P_s(\boldsymbol{x}, y) = P_s(\boldsymbol{x})\ P_s(\mathrm{MB}_s(Y) = \boldsymbol{m}). \tag{7.2}$$

We apply a shuffling scheme to balance the conditional distribution of Y by switching Y's values in the data stream. For every $\mathrm{MB}_s(Y) = \boldsymbol{m}$ and $Y = y_i$,

- if $P_s(y_i \mid \boldsymbol{m}) \leq 1/\|Y\|$, we keep y_i unchanged;
- otherwise, we switch some y_i to y_j, where $P_s(y_j \mid \boldsymbol{m}) \leq 1/\|Y\|$, with transition probabilities. The probability to keep y_i unchanged is $\frac{1/\|Y\|}{P_s(y_i|\boldsymbol{m})}$. For the remaining part, the probability of changing y_i to y_j is proportional to $1/\|Y\| - P_s(y_j \mid \boldsymbol{m})$.

We have the obfuscated distribution $P_t(y_i \mid \boldsymbol{m}) = 1/\|Y\|$ for all y_i and $\boldsymbol{m}$. Therefore, it makes Y independent of $\mathrm{MB}_s(Y)$. Theorem 7.4.2 in 7.4.1 proves that our obfuscation method is sufficient to ensure the independence between $\boldsymbol{X}$ and Y for any Bayesian network.

Online updating for obfuscation schemes: The obfuscation schemes are generated according to $P_r(\boldsymbol{X}, Y)$. However, $P_r(\boldsymbol{X}, Y)$ is a part of the unknown model to learn from the data. We break this dilemma by designing an online approach which estimates the probability and updates the scheme continuously.

First, MORSE initializes an estimated $P_r(\boldsymbol{X}, Y)$ with a random distribution. Then it updates the probability at run-time from the collected data stream. The online procedure is secure as long as initial $P_r(\boldsymbol{X}, Y)$ is unknown to the attacker.

In detail, MORSE splits the time into separated periods. In each time period, the security scheme is unchanged. To update the estimation from the collected data, we need to derive the distribution before sampling and shuffling. Notice that, we focus on reconstructing the

Bayesian model, namely the conditional probabilities instead of the actual data. eventually converges to an accurate estimate.

The sampling and shuffling in essence provide a mapping between the original distribution and observed distribution. We use transition matrix $T_{\boldsymbol{m}}$ to denote the transition probabilities when $\mathrm{MB}_s(Y) = \boldsymbol{m}$. To reconstruct the model, we first derive the probability *before* shuffling, which equals to multiplying the inverse of transition matrix $T_{\boldsymbol{m}}$ with the distribution after shuffling. We will show that $T_{\boldsymbol{m}}$ is invertible in §7.4.3 and its inverse multiplies by the final observed distribution yields the post-sampling distribution. Next, among all groups where sampling is possible, we derive the probability before sampling steps in turn by dividing the distribution by sampling probability and then taking normalization.

After estimating the distribution, we update the sampling and shuffling scheme based on the probability estimation with approaches described in §7.3.1 and §7.3.2. We will prove that, this iterative procedure guarantees the estimation of distribution converges to the pre-sampling distribution. Eventually, the model can be correctly reconstructed albeit obfuscated.

7.4 Analyzing MORSE

We first show that MORSE is secure against both the basic attacker and advanced attacker. Then we formally prove the reconstructability and convergence.

7.4.1 Security Against Basic Attackers

We first prove that the model is obfuscated against a basic attacker. This type of attacker directly uses the observed data for model training. We first show that sampling can help each group achieve variable independence in Theorem 7.4.1. Afterwards, Theorem 7.4.2 proves that all remaining variables will achieve independence after shuffling. Therefore, all $\boldsymbol{X}$ and Y are independent after sampling and shuffling.

Theorem 7.4.1. Sampling in each group eliminates the dependencies within the group and does not affect the relationship outside the group.

Proof. To prove this theorem, we calculate the probability distribution after sampling.

The obfuscation scheme applies sampling probability $S_V\left(V = v, \text{pa}(V) = \boldsymbol{v}_{pa}\right)$ to the original conditional distribution $P_r(V = v \mid \text{pa}(V) = \boldsymbol{v}_{pa})$. The result is $P_r(v \mid \boldsymbol{v}_{pa}) \cdot S_V(v, \boldsymbol{v}_{pa}) = 1/\|V\|$, which is a constant number for all conditions of V and $\text{pa}(V)$. Hence, $P_s(V = v) = 1/\|V\|$ replaces $P_r(V \mid \text{pa}(V))$ in the joint distribution function after sampling. The dependency between V and $\text{pa}(V)$ is eliminated in the new Bayesian network.

Additionally, all conditional probability distributions other than $P_s(V \mid \text{pa}(V)$ keep the same after sampling with group $\{V, \text{pa}(V)\}$, therefore, the other dependencies in the Bayesian network remain unchanged. □

Corollary. If all groups with Y do not have 0 probabilities in their conditional distribution functions, sampling is applicable to all the groups, and Y is independent to $\boldsymbol{X}$ after sampling.

We next prove that $\boldsymbol{X}$ and Y are always independent after sampling and shuffling.

Theorem 7.4.2. Output variable, Y, is always independent of input variables, $\boldsymbol{X}$, after MORSE's obfuscation.

Proof. We first prove that the shuffling scheme only needs to handle Markov blanket $\text{MB}_s(Y)$ of Y after sampling. We then show this is sufficient to make Y independent of $\text{MB}_s(Y)$.

The property of Markov blanket [NSL13] shows that the other variables are independent of Y conditioned on $\text{MB}_s(Y)$. Since sampling eliminates the dependencies, the remaining groups after sampling construct the Y's new Markov Blanket $\text{MB}_s(Y)$. The probabilities after sampling are

$$P_s(\boldsymbol{x}, y) = P_s(\boldsymbol{x})P_s(y \mid \boldsymbol{x}) = P_s(\boldsymbol{x})P_s(y \mid \text{MB}_s(Y) = \boldsymbol{m}).$$

Next, we prove MORSE manipulates the conditional dependency and eliminates the dependency in $P_s(Y \mid \mathrm{MB}_s(Y))$. §7.3.2 shows that the sampling scheme for each condition of the Markov blanket $\mathrm{MB}_s(Y) = \boldsymbol{m}$ has a transition matrix $T_{\boldsymbol{m}}$ that balances the obfuscated model's distribution. $T_{\boldsymbol{m}}$ is a $\|Y\| \times \|Y\|$ matrix. Element at row i and column j of $T_{\boldsymbol{m}}$ represents the probability of switching Y from ith value to jth. If $P_s(Y = y_i \mid \mathrm{MB}_s(Y) = \boldsymbol{m}) \leq 1/\|Y\|$, we keep the post-sampling records here since they are below or equal to the average. Otherwise, we divide the extra samples to the ones below average by changing the value of Y.

The transition probabilities change $P_s(Y \mid \mathrm{MB}_s(Y) = \boldsymbol{m})$ to a uniform distribution that $P_t(y) = P_t(y \mid \boldsymbol{m}) = 1/\|Y\|$ for all y. Since the shuffling scheme only changes Y's value, $P_s(\boldsymbol{X})$ is not affected. $\boldsymbol{X}$ and Y are independent after the obfuscation. □

7.4.2 Security Against Advanced Attackers

We consider the advanced attacker who is aware of MORSE's system, except the parameters for model obfuscation. We show that such an attacker cannot estimate the correct model given the obfuscated observation.

Theorem 7.4.3. An attacker cannot infer the original model from the obfuscated observation from MORSE.

Proof. Suppose sampling is applied to group $\{V, \mathrm{pa}(V)\}$, any distribution $P'_r(V \mid \mathrm{pa}(V))$ with probabilities all greater than 0 can generate the same post-sampling distribution as the original $P_r(V \mid \mathrm{pa}(V))$. Suppose shuffling is applied to Y and its Markov Blanket $\mathrm{MB}_s(Y)$, distribution $P'_s(Y \mid \mathrm{MB}_s(Y))$ with arbitrary probabilities can generate the same post-shuffling distribution as the real $P_s(Y \mid \mathrm{MB}_s(Y))$.

Recall Theorem 7.4.1 that the sampling is feasible for any conditional distributions that do not have 0 probabilities, and all the post-sampling conditional probabilities equal to $1/\|V\|$.

So, for each sampling step with group $\{V, \text{pa}(V)\}$, any initial conditional distribution $P'_r(V \mid \text{pa}(V))$ without zero-probabilities can generate the same post-sampling conditional distribution, $P_s(v \mid \boldsymbol{v}_{pa}) = 1/\|V\|$.

Additionally, the sampling process does not change the conditional distributions outside the group $\{V, \text{pa}\, V\}$. So the post-sampling distributions, $P_s(V \mid \text{pa}(V))$ from the true model and $P'_s(V \mid \text{pa}(V))$ from the randomly generated model, are the same.

Similarly, the shuffling module generates the same observations from different post-sampling distributions. Therefore, the attacker cannot infer the original model without knowing the obfuscation schemes. □

The theorem shows that an advanced attacker cannot estimate the distributions before either sampling or shuffling. Since they are processed independently, all the sampling probabilities and transition matrices are indispensable for reconstructing the model. Infinitely many distributions can generate the observation with similar probability (validated in §7.6.2). Therefore, the original model is secure against advanced attackers.

7.4.3 Reconstructability and Convergence

This section shows that MORSE is invertible, and the reconstructed distribution converges to the real distribution. We first show that we can reconstruct the real distribution regardless of the sampling and shuffling scheme in Theorem 7.4.4.

Theorem 7.4.4. Original distribution can be reconstructed from the post-shuffling distribution with the knowledge of the sampling probabilities and shuffling transition matrices.

Proof. We show that both sampling and shuffling schemes are invertible so that the original distribution can be reconstructed.

$T_{\boldsymbol{m}}$ is the transition matrix defined in §7.3.2. Since there is no loop for switching the value of Y, $T_{\boldsymbol{m}}$ can be written as a triangular matrix with all diagonal values greater than 0.

Thus, the transition matrix and shuffling process are invertible. By multiplying the observed distributions with $T_{\boldsymbol{m}}^{-1}$, we can recover the distribution before shuffling.

Second, we calculate the original distribution from the post-sampling distribution. Multiplying and scaling the distribution with the inverse of the sampling rates generate the original distribution. The expectation of the reconstructed distribution $\hat{P}_r(\boldsymbol{X}, Y)$ equals to $P_r(\boldsymbol{X}, Y)$. □

Corollary. The estimated distribution with the online method converges to the real distribution.

Proof. From Theorem 7.4.4, the expectation of the reconstructed distribution is the real distribution in any iteration. Based on the law of large numbers (LLN), since we update the estimation with the average of the results $\hat{P}_r(\boldsymbol{X}, Y)$ obtained from a large number of iterations, it eventually converges to the expectation, which is the real distribution. □

7.4.4 Discussion

Obfuscating other models: Though we focus on obfuscating the Bayesian models in MORSE, the data-dependent sampling can be generalized in the follow-up works. In §7.6.2, we show that MORSE still protects the input-output relation when the attacker applies other machine learning methods.

Multiple outputs: MORSE could be extended to multiple-output cases without major modification. We repeat the procedure of deriving sampling/shuffling schemes for every output sequentially. Note that processing one output may change the Bayesian network structure. It updates the structure when deriving scheme for the later outputs.

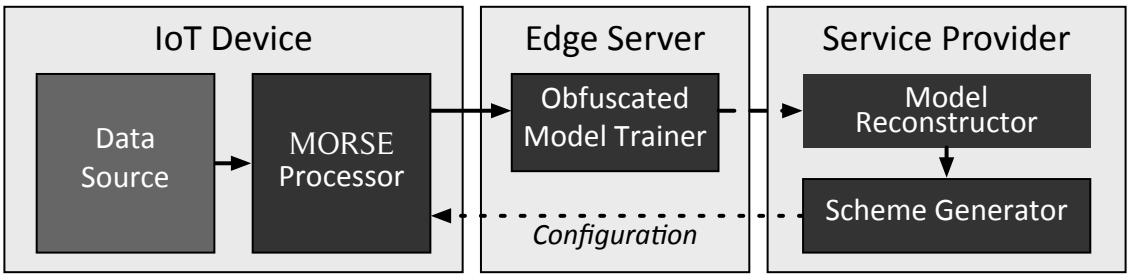

Figure 7.4: Implementation of MORSE.

7.4.5 Applications

We present two real-world applications in healthcare and automotive that MORSE can be used.

Preventive medical diagnosis: Home tele-health services collect metrics from medical IoT devices to enable preventive diagnosis by a remote doctor. For example, sports injury prevention screenings are done by the patient completing several fundamental movement patterns and may measure angle, distance, or offset of completion of patterns.

Vehicle reliability diagnostics: Car manufacturers install multiple in-car sensors to monitor cars' reliability state. The task is to track multiple sensor readings and reason which sensor reading(s) is the most probable cause of the reliability issue. The reliability diagnosis includes normal tear, engine issue, and tire issue. Different sensors could be scattered across the factory, which are first aggregated in a local gateway and then uploaded for further learning tasks.

7.5 Implementing MORSE in an IoT system

In this section, we describe MORSE system implementation, which is shown in Fig. 7.4. We build MORSE as software modules in IoT devices, edge servers, and service providers. As software-based implementations, they are applicable to most IoT systems. The sampling and shuffling are implemented in MORSE Processor at the IoT device, while the service

provider reconstructs the authentic model and updates the MORSE scheme based on it. The communication between modules is realized via (local) inter-process communication and (remote) socket communication.

• *MORSE processor on IoT devices:* We implement MORSE processor in the IoT devices. It fetches the sampling and shuffling scheme from the service provider, samples and shuffles data from different sensors, and then uploads the processed data to the edge server. The data is processed efficiently both in time and space. To improve the look-up efficiency, we use in-memory hash tables to store the sampling probabilities and transition matrices.

• *Obfuscated model trainer at the edge server:* The edge saves all incoming data streams in its database. After receiving a batch of data, it learns the (obfuscated) Bayesian model with package `bnlearn` [Scu09].

• *Reconstructor at service provider:* The reconstructor takes the obfuscated model and estimates the real model, as shown in §7.3.2. The mapping is a sequence of matrix calculations, which we implement using Python package `numpy.linalg`. It is then used to update shuffling and sampling schemes.

• *MORSE scheme generator at service provider:* MORSE generates our protection scheme in §7.3 - §7.3.2. This scheme is then forwarded to IoT devices securely with encryption. We leverage Python linear algebra libraries to generate the sampling and shuffling scheme. Since the scheme update happens infrequently as the data is processed in batches, the overhead incurred at the scheme generator is less important in MORSE.

7.6 Evaluation

7.6.1 Experiment Setup

Testbed: We build a testbed as shown in Fig. 7.5. A Beaglebone Green board with an AM3358 1GHz CPU and 512MB RAM and a Raspberry Pi 3 with a Cortex-A53 1.2GHz

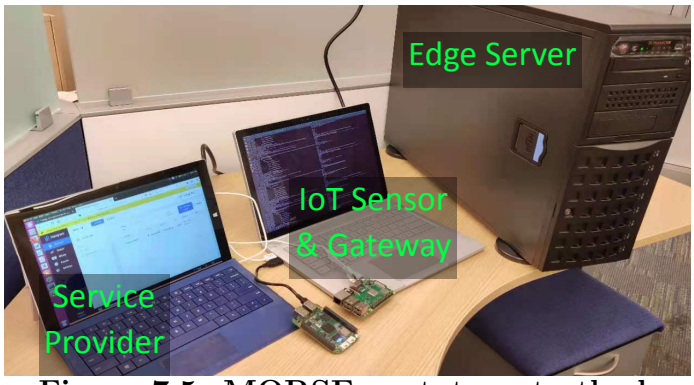

Figure 7.5: MORSE prototype testbed.

Dataset	Nodes	Edges	Parameters	Output (Y)
SACHS	11	17	178	`Akt`
CANCER	5	4	10	`Cancer`
ALARM	37	46	509	`LVV`

Table 7.3: Characteristics of the datasets for evaluation.

CPU and 1GB RAM serve as the sensor and gateway. We write a program that feeds the sensor datasets locally to the gateways. They are connected to a Microsoft Surface Book with a 2.4GHz Intel Core i5-6300U and 8GB RAM for synchronization. It establishes a TCP connection with the edge server. We use a server with a 12-core Intel Xeon Silver 4214 CPU and 32GB RAM. A Microsoft Surface2 with an Intel Atom x7 CPU and 4GB RAM serves as the service provider.

Application and dataset: We examine the performance of MORSE with three IoT-related applications: SACHS [SPP05], CANCER [LS88], and ALARM [BSC89], from different fields. They have gold standard Bayesian network models from the previous studies [KN10, GFZ19], which cover different scales and structures. Table 7.3 summarizes their parameter settings. We extract 100,000 records for training (including obfuscating) and 10,000 records for testing in each dataset.

Evaluation metrics: We evaluate our design with the following metrics: mutual information [CT12] (for measuring input-output mutual dependencies), accuracy (for measuring the prediction performance), and Euclidean distances (for comparing the similarity between the reconstructed model and the original model). Additionally, we design a novel metric $RAUC$

(Relative Area Under Curve Score) to quantify the similarity among the obfuscated model, the original model, and the uninformed model.

We devise $RAUC$ to quantify the security score of a modified model comparing with the original. $RAUC$ is calculated with

$$RAUC(M_{\text{mod}} \mid M_{\text{real}}) = \left| \frac{AUC(M_{\text{mod}}) - AUC(M_{\text{uninf}})}{AUC(M_{\text{real}}) - AUC(M_{\text{uninf}})} \right|, \tag{7.3}$$

where $AUC(M_*)$ denotes the AUC score of model M_*. We take the average AUC score when Y has more than two classes [HT01]. M_{real}, M_{mod}, and M_{uninf} correspond to the real, obfuscated/reconstructed, and uninformed models. The uninformed model's AUC is always 0.5. $RAUC$ quantifies the relative likeness between the modified model and the original/uninformed model. If the modified model performs the same as the original model, its $RAUC$ is 1. If the model works as if taking random guessing (uninformed classifiers), its $RAUC$ equals 0.

7.6.2 Security of MORSE

We empirically that MORSE is secure against both basic and advanced attackers.

Input-output independence: We used the mutual information between the inputs and the output, $I(\boldsymbol{X}; Y)$, to measure the independence. Fig. 7.6a plots the relationship between mutual information and the number of data records. By applying our obfuscation approach, the input-output mutual information in the collected data converges to 0.

Obfuscating Bayesian model for basic attackers: Fig. 7.7 shows that the models directly learned from processed datasets have poor performance in the classification tasks. We repeat experiments 20 times for each dataset, averaging out the randomness introduced by probabilistic sampling and shuffling. Moreover, $RAUC$ scores for the Bayesian models shown in Table 7.4 are all close to 0 after obfuscation, which equal to 5.73%, 2.80%, and 0.60%, respectively. Therefore, the obfuscated model is close to an uninformed model which

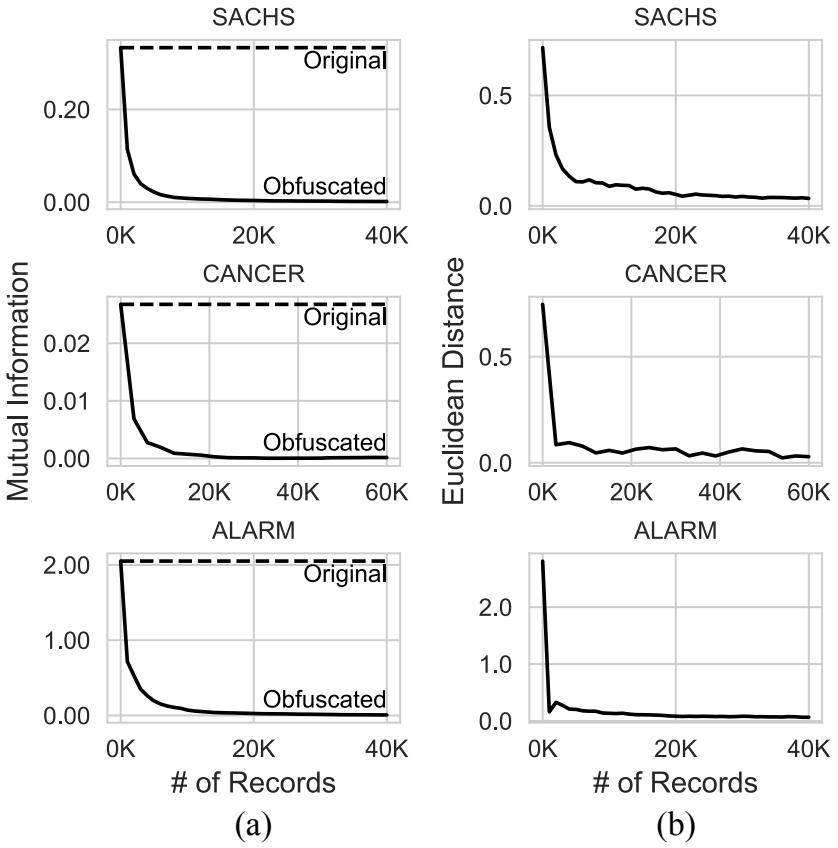

Figure 7.6: Input-output independence and correctness of reconstruction.

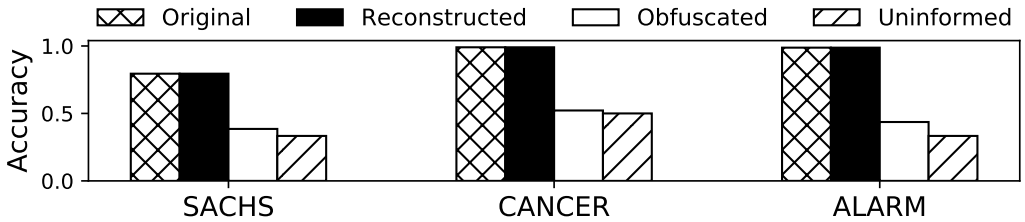

Figure 7.7: High accuracy of reconstructed model on different datasets after convergence.

Dataset	Bayesian Network (BN)		Other Models			
	Obfuscated	Reconstructed	NB	RF	GB	NN
SACHS	5.73	99.71	4.39	4.39	0.99	4.67
CANCER	2.80	97.82	5.85	2.33	3.32	4.61
ALARM	0.60	99.15	5.75	0.78	2.87	1.79

Table 7.4: *RAUC*s(%) of MORSE and effectiveness against other machine learning models.

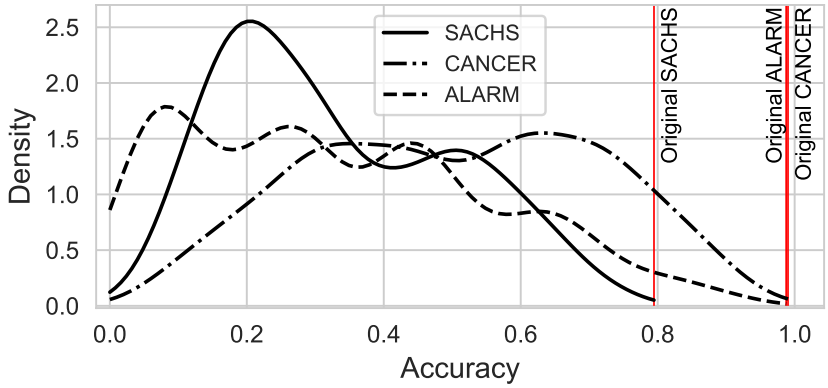

Figure 7.8: Accuracy distribution of possible models learned by an advanced attacker.

randomly guesses Y and the basic attacker can learn nothing.

Obfuscating Bayesian model for advanced attackers: We let an advanced attacker generate 2000 models according to the observation for each dataset. Fig. 7.8 shows the accuracy distribution tested on these models. The average classification accuracies are 0.333, 0.503, and 0.334 for SACHS, CANCER, and ALARM, i.e., the model reconstructed by an advanced attacker is similar to the uninformed one in expectation.

Since the input and output are independent after the obfuscation, MORSE is still effective when the attacker applies other machine learning models to learn the input-output relation. Table 7.4 shows that if we learn other models from the observed data, such as naïve Bayes (NB), random forest (RF), gradient boosting (GB), and feed-forward neural network (NN) classifiers, the $RAUC$ scores of them are still close to 0.

Approach	Encrypt/Process (s/1000 records)	Decrypt/Reconstruct (s/1000 records)	Scheme Update (s/update)
Paillier	601.1	181.1	-
AES-CBC	0.406	0.452	-
AES-CTR	0.391	0.328	-
MORSE-Naïve	0.641	0.257	0.208
MORSE	**0.039**	**0.008**	**0.044**

Table 7.5: Comparison of processing latency between encryption and MORSE.

7.6.3 Correctness of Model Reconstruction

Convergence of the reconstructed model: we measure the Euclidean distance between the reconstructed model and the original model to show the correctness. It is shown in Fig. 7.6b as the distance quickly approaches 0 after a few thousand original data points.

Performance of the correctly reconstructed model: We show the performance of the reconstructed model with accuracy and $RAUC$. Fig. 7.6b and Table 7.4 demonstrate that the reconstructed models produce similar accuracy and $RAUC$ compared to the original models.

7.6.4 Overhead

Processing time overhead: MORSE obfuscates the model with high efficiency. We evaluate the average sampling and reconstruction latency compared to other common encryption methods. We choose AES-CBC from [FGK03] and AES-CTR from [MS10] as encryption baseline and Paillier from [Pai99] as homomorphic baseline. We compare them on ALARM.

The results are shown in Table 7.5. MORSE runs 10X faster for data processing and 40X for reconstructing model (data decryption for AES) compared to AES baselines. MORSE is a few magnitudes faster than the homomorphic approach, which could overwhelm the resource-constrained IoT devices. We also compare MORSE with a naïve data-dependent sampling solution which samples and shuffles data based on *all* variables. This solution runs even slower than AES, which shows the necessity and the effectiveness of MORSE design.

7.7 Related Work

Model extraction attack is a close topic to ours. Existing works investigate several attack models that steal information of models. [LEM18] introduces a method to steal the hyper-parameters from machine learning models. [JSM19, JCB20] show different approaches to extract the DNN models from the prediction APIs. These attacks target machine learning as a service (MLaaS) platform. The attacker in our setting has direct access to the data on the edge server. Also, to the best of our knowledge, this is the first work addressing model attacks and protection in the IoT sensor context.

Several works have studied detecting and defending against the model learning attacks. [JSM19, CCW20] propose ways to detect the model extraction behaviors and hold services whenever they identify the attack, which is not suitable in our settings since our sensor data always go through the edge server. To prevent the adversary from accessing the original data, [GDL16] offers a way to transform the trained neural network to be used on encrypted data. [AEH15] introduces a method with a fully homomorphic function for learning random forest or naïve Bayes models. [HZX18] protects user-level computations from untrusted privileged cloud services with hardware-specific enclaves. Our work differs from their work in two aspects. Firstly, we focus on securing the *probabilistic* models, which cannot be protected by these works. Second, MORSE does not require complex (homomorphic) encryption or hardware features while protecting the model. This is suitable for capacity-constraint IoT devices.

Protecting IoT data from the edge/cloud server's access is a popular topic. [SBH17, SBR20, CBB19, HZX18] secure IoT data access with decentralized authentication. [SHD15, SHB17] leverage homomorphic encryption to prevent the cloud from obtaining the original data. In contrast, MORSE focuses on directly obfuscating the model rather than individual data or particular queries. Sampling provides a fresh yet efficient way for IoT security without expensive operations.

7.8 Conclusion

In this work, we study the *model* protection problem in the IoT setting where the model is learned from the sensor/output data. A novel data-dependent sampling scheme is proposed. The scheme produces the dataset that obfuscates the dependency which is critical to model learning. Meanwhile, the real model can still be reconstructed from the obfuscated model. To the best of our knowledge, MORSE is the first work to explore the possibility of such a model-centric security design for a class of IoT scenarios. We hope MORSE could stimulate more efforts towards model security for IoT usages.

CHAPTER 8

Conclusion and Future Work

As we enter the concluding phase of this dissertation, it's important to underscore the vital communication chasm that exists between the Deaf and hearing communities. Despite strides made in technology and communication tools, there currently lacks a comprehensive end-to-end solution that effectively bridges this gap. Existing approaches either fall short in accessibility or capacity, leaving a void in truly seamless communication. However, with recent advancements in hand and body detection algorithms, devices now have a chance to "see" and interpret sign language in a manner similar to humans. Capitalizing on this promising trend, our research has centered around developing Assistive AR system solutions to effectively bridge this communication divide.

A significant breakthrough in our journey to connect sign and oral language can be traced back to William Stokoe's groundbreaking theory, which posits sign language as a natural language replete with its own phonetic and grammatical nuances. Embracing this principle has allowed us to apply linguistic studies of ASL to automatic sign translation and mobile systems, opening up exciting avenues of accessibility and communication.

The main claim of this dissertation is that the gap between Deaf and hearing communication can be bridged through a series of systematic solutions. First and foremost, we honed in on urgent needs scenarios, such as emergency calls, and discovered that translation could be accomplished by categorizing signs into sign parameter groups. We also found that ASL possesses its own unique syntax rules, differing from English but just as systematic. Building on this understanding, we broadened our focus from emergency scenarios to more general-

ized communication contexts by transforming ASL dictionaries into machine-interpretable parameter sets.

Recognizing that languages, including sign languages like ASL, are dynamic and vary across time, regions, and sub-cultural groups, we prioritized continuous learning and collaboration with sign language users. To this end, we developed the AnySign platform. This comprehensive solution provides ASL search functionality, sign collection, full sentence translation, and interactive features, offering a promising path towards true communication inclusivity.

In the ensuing sections, we will succinctly recap our findings in Section §8.1, share insights gleaned from our research in Section §8.2, and explore potential avenues for future research in Section §8.3.

8.1 Summary of Results

This dissertation illuminates the complexities of communication barriers faced by the deaf community in an interconnected society, largely resulting from a blend of factors such as lack of accessibility, inadequate support for sign languages, and a shortage of efficient translation resources. To alleviate these issues, we propose a novel approach with Assistive Augmented Reality (AR) to bridge the divide between the deaf and hearing communities. This encompasses the creation of Sign-to-911 for emergency situations, a general sign language translation mechanism, and AnySign, a comprehensive cross-device platform. By utilizing AR as an innovative bridge for communication, we provide robust, accessible, and efficient solutions. The principal contributions of this dissertation are as follows:

Sign-to-911: Emergency Call for Sign Language Users with Assistive AR Recognizing the urgent need for accessible and efficient sign language support during emergency situations, we developed *Sign-to-911*. This system is a rapid, lightweight solution specifically

designed to bridge the ASL-English communication gap during emergencies. By integrating AI and ML models with specific ASL linguistic domain knowledge, we successfully reduced model complexity while maintaining high translation accuracy and speed. Rigorous evaluations with real ASL users and stress tests in simulated emergency situations confirmed its effectiveness and reliability. Importantly, *Sign-to-911* is designed to function independently on mobile and wearable devices without reliance on cloud or edge support, ensuring it can be of service under any circumstances. It represents a significant stride in emergency communication for the ASL user community, potentially serving as a lifesaver in critical moments.

AnySign: A Comprehensive Cross-device Sign Language Interaction Platform To address the unique communication challenges faced by the Deaf community, we introduced *AnySign*, a comprehensive platform designed to enhance the general accessibility of ASL. AnySign brings together the *Sign Dictionary*, *English to ASL Translator*, and *Teach-Me-Sign* modules to provide a comprehensive ASL learning and communication platform. One of the highlight features of the platform is *SignChat*, an AI-powered chatbot capable of enabling a new mode of ASL human-machine interaction. Feedback from system users demonstrates the potential of *AnySign* to effect change and catalyze further advancements in ASL communication.

MORSE: Private Model Protection in Machine Learning Pipeline In response to the increased privacy and security concerns associated with deploying machine learning models on wearable and IoT devices, particularly in the context of Assistive AR, we devised *MORSE*, a lightweight model privacy protection solution. MORSE employs a novel sampling approach that effectively protects the privacy of models without exposing sensitive personal information to cloud platforms. This approach ensures both functionality and trainability of models while maintaining the benefits of AI and machine learning. Importantly, MORSE is developed with a focus on IoT and wearable applications, demonstrating our commitment

to secure and trustworthy computing in this emerging field where user privacy is paramount. The integration of MORSE into Assistive AR devices could offer users confidence in the secure use of AI technologies, making a significant step toward the broader adoption of these tools for improved communication and accessibility.

8.2 Lessons Learned

This dissertation journey was a critical platform for learning and understanding, not only from a technological perspective but also from a socio-cultural standpoint. Key lessons that we gleaned include the importance of cultural respect and the need for interdisciplinary collaboration.

8.2.1 Acknowledging and Embracing Cultural Variance

Acknowledging and valuing cultural differences, notably within the diverse Deaf community, was a cornerstone of our research. In a world where hearing individuals often dictate discourse, there is a tendency among tech developers to focus on what they already possess, potentially neglecting the specific needs of Deaf individuals. The guiding question that underpinned our dissertation was, "Is this truly beneficial to the Deaf community?"

Our platform, christened as AnySign, embodies this commitment to creating sign language accessibility at any moment, in any place. But the name conveys a dual message: it's an affirmation saying "I need sign", encouraging the wider community to respect and employ sign languages equivalently to oral languages, fostering greater inclusivity in our rich and dynamic world.

8.2.2 The Crucial Role of Interdisciplinary Collaboration

The second key takeaway from our research journey stresses the indispensability of interdisciplinary collaboration in this arena. Misinterpretations of sign language have remained entrenched, with prevalent assumptions incorrectly reducing it to merely a physical manifestation of English. To thoroughly grasp the complexities and nuances of sign languages, debunking such misapprehensions is critical – a task best accomplished through concerted interdisciplinary collaboration.

By bridging the divides between various fields – including linguistics, technology, and social studies – we can collaboratively dispel longstanding myths and misconceptions about sign languages. This collaborative effort is not only vital for nurturing a more comprehensive understanding of sign languages but also instrumental in developing more precise and effective communication tools. Essentially, these concerted efforts across disciplines are the catalyst for innovative breakthroughs, paving the way towards more sensitive and effective communication technologies. Our collective goal is to better cater to the Deaf community, fostering an inclusive society that values and embraces linguistic diversity.

8.3 Looking Forward

8.3.1 Harnessing AIGC for Streamlined Sign Language Captioning

In today's digital age, movies, and online video platforms serve as vital gateways for knowledge acquisition. Offering sign language captions can increase linguistic accessibility for individuals who prefer sign languages, such as deaf children and native sign language users. Recently, companies like Disney have acknowledged this need, implementing human interpreters for select content [Sha21, Chi21]. However, due to the scarcity of certified ASL interpreters and the continuous eruption of new online content, manual interpretation for all these media remains an unrealistic task.

One potential solution lies in automated caption generation, akin to YouTube's approach, where speech is understood and transcribed into sign language. Central to this process is the translation from English to ASL while adhering to accurate syntax and sign usage. In this work, we have explored the viability of generating ASL with its unique grammar and signs.

However, unlike text, vocal communication encompasses richer information encapsulating the speaker's tone and mood. To capture this nuance in sign language, modern deep learning models such as the Generative Adversarial Network (GAN) [GPM20] framework and stable diffusion methods [RBL22, ZA23] show promise in producing photorealistic sign language interpretations.

Looking ahead, we foresee a promising future where the deployment of AI-Generated Content (AIGC) could significantly enhance sign language captioning. Leveraging AIGC not only promises to widen the accessibility of this crucial form of communication but also enhance comprehension, particularly benefiting children and sign language users.

To put this theory into practice, we undertook a preliminary experiment where image generation was appended to the ASL production pipeline. Using the stable diffusion model, we were successful in converting avatar-rendered images into model-generated human-like images. While this trial yielded encouraging results in terms of individual image generation, two significant challenges emerged: 1) The lack of control over the shape of generated hands sometimes resulted in deformed appearances—a flaw which is unacceptable in sign language production; 2) Maintaining content continuity was a problem, given the fluctuations in clothing and facial features over time. Despite these challenges, we remain confident about the potential of AIGC in revolutionizing sign language captioning.

8.3.2 Leveraging Transformers for an Advanced ASL Translation Pipeline

The task of translating sign language involves not just sign recognition but also the interpretation of sign sequences to construct a meaningful sentence. Currently, the approaches explored in our translation system depend on segmenting signs based on the hold states of hands. However, in natural sign language expression, signers possess differing signing intervals, introducing complexity into the segmentation process.

This variability calls for a more adaptable solution. Sequence-to-sequence models, such as transformers [VSP17], have been shown to effectively handle similar challenges in the field of Natural Language Processing. They have the potential to do the same for sign language translation. Transformers' attention mechanism allows them to focus on different parts of the input sequence when generating the output, enabling them to handle variable length and order, which is crucial for interpreting sign sequences.

Moreover, transformers' ability to process sequences in parallel rather than in a sequential manner can lead to a significant reduction in computation time. In the context of sign language translation, this capability can result in real-time translation services that are essential in many practical applications.

However, this approach does entail more comprehensive data collection and pre-processing. Given the inherent complexity of sign languages, obtaining high-quality, varied data for model training is essential. Additionally, the pre-processing of sign language data, such as proper segmentation and feature extraction, will be critical in ensuring the effective use of the transformer model.

Therefore, employing transformer models within the ASL translation pipeline could be a promising path to enhance the robustness and efficiency of sign language translation systems. While this approach demands extensive data collection and pre-processing efforts, it holds the potential to make ASL translation more accessible, versatile, and practical. In the long run, these advancements could contribute significantly to bridging the communication gap

between the Deaf and hearing communities.

8.3.3 Assistive AR for a Broad Spectrum of Applications

As the world embraces the era of augmented reality, Assistive AR presents enormous potential to extend beyond just aiding sign language communication, offering assistance to a much broader range of use cases.

Enhancing Communication for Hearing Impairment. One such application is in aiding those grappling with hearing loss, which is often due to aging. Traditional solutions such as hearing aids or cochlear implants (CI) come with their own set of challenges - they often demand a significant period for users to adjust to, and even after that, they might not restore the hearing ability completely. Assistive AR can offer an instant and intuitive solution to this challenge, by offering real-time captioned text, hence enhancing the communication experience for those with hearing loss.

Assistive AR for Low-vision Support. In the realm of low-vision assistance, Assistive AR offers promising avenues. Features such as object detection and content reading can be provided by Assistive AR technology, allowing individuals with low-vision to navigate their surroundings more effectively and safely. By presenting enhanced visual cues and magnified or simplified visual information, Assistive AR holds the potential to significantly improve the quality of life for people with visual impairments.

Enriching Daily Life Activities with Assistive AR. Beyond these specific applications, Assistive AR holds vast potential for enriching general daily life activities. For instance, during sporting activities, Assistive AR could offer real-time data and feedback to athletes, enhancing performance and training effectiveness. Similarly, in navigation, Assistive AR could offer more intuitive and immersive wayfinding solutions, transcending what traditional GPS or map services offer.

In conclusion, the prospective applications of Assistive AR, while vast, present a particularly promising tool for bridging the divide between the Deaf and hearing communities. By redefining communication boundaries and fostering inclusivity, Assistive AR plays an instrumental role in making the world more accessible for everyone. Beyond this vital contribution, Assistive AR also holds immense potential in revolutionizing daily life activities, from aiding those with visual or hearing impairments to enhancing athletic performance and navigation. In essence, this technology promises a future wherein AR integrates seamlessly into our everyday lives, breaking barriers, enhancing experiences, and ensuring that no one is left behind. The evolution of Assistive AR, as evidenced in this dissertation, is not just a thrilling prospect for technological progression, but also a beacon of hope for a more inclusive and enhanced future for all.

REFERENCES

[AC98] Alan Agresti and Brent A Coull. "Approximate is better than "exact" for interval estimation of binomial proportions." *The American Statistician*, **52**(2):119–126, 1998.

[Acc] "Accessibility Service." `https://developer.android.com/reference/android/accessibilityservice/AccessibilityService`. Accessed: 2022-11-29.

[ada90] *Americans with Disabilities Act*. U.S. Department of Justice, 1990.

[AEH15] Louis JM Aslett, Pedro M Esperança, and Chris C Holmes. "Encrypted statistical machine learning: new privacy preserving methods." *arXiv preprint arXiv:1508.06845*, 2015.

[AGR14] Sílvia Grasiella Moreira Almeida, Frederico Gadelha Guimarães, and Jaime Arturo Ramírez. "Feature extraction in Brazilian Sign Language Recognition based on phonological structure and using RGB-D sensors." *Expert Systems with Applications*, **41**(16):7259–7271, 2014.

[And] "Andronix App." `https://andronix.app/`. Accessed: 2023-03-09.

[And23a] "Android Camera API." `https://developer.android.com/guide/topics/media/camera`, Mar. 2023.

[And23b] "Android (Go edition)." `https://www.android.com/versions/go-edition/`, Mar. 2023.

[And23c] "Android MediaCodec." `https://developer.android.com/reference/android/media/MediaCodec`, Mar. 2023.

[And23d] "Android Motion sensors." `https://developer.android.com/guide/topics/sensors/sensors_motion`, Mar. 2023.

[And23e] "Android TextToSpeech." `https://developer.android.com/reference/android/speech/tts/TextToSpeech`, Mar. 2023.

[ANO01] C. Andersson, I. Nilsson, P. Olsson, G. Slot, J. Sörensen, D. Liechty, R. Hunter, S. Kambhatla, S. Adermann, Camp M., P. Godia-Caner, R. Mettälä, and M. Box. "RFCOMM with TS 07.10.", 2001.

[aod05] "Accessibility for Ontarians with Disabilities Act.", 2005.

[Ars15] ArsTechnica. "https://arstechnica.com/tech-policy/2015/04/att-fined-25-million-after-call-center-employees-stole-customers-data/.", 2015.

[ASL23] "ASL Translator." https://apps.apple.com/us/app/asl-translator/id421784745?correlationId=fc9f5193-5430-4cf1-8249-0b7052ee005c, Mar. 2023.

[AZZ18] Mohamed Aktham Ahmed, Bilal Bahaa Zaidan, Aws Alaa Zaidan, Mahmood Maher Salih, and Muhammad Modi bin Lakulu. "A review on systems-based sensory gloves for sign language recognition state of the art between 2007 and 2017." *Sensors*, **18**(7):2208, 2018.

[b11] "Metaverse 3D Avatar Creator — Ready Player Me." Accessed: 2023-02-21.

[b820] "MediaPipe Holistic — Simultaneous Face, Hand and Pose Prediction, on Device.", 12 2020. Accessed: 2023-02-21.

[b9] "ARKit — Apple Developer Documentation." Accessed: 2023-02-21.

[Bat78] Robbin Battison. "Lexical borrowing in American Sign Language." 1978.

[BGR20] Valentin Bazarevsky, Ivan Grishchenko, Karthik Raveendran, Tyler Zhu, Fan Zhang, and Matthias Grundmann. "Blazepose: On-device real-time body pose tracking." *arXiv preprint arXiv:2006.10204*, 2020.

[BH20] Maryam Bandukda and Catherine Holloway. "Audio AR to support nature connectedness in people with visual disabilities." In *Adjunct Proceedings of the 2020 ACM International Joint Conference on Pervasive and Ubiquitous Computing and Proceedings of the 2020 ACM International Symposium on Wearable Computers*, pp. 204–207, 2020.

[BKL09] Steven Bird, Ewan Klein, and Edward Loper. *Natural language processing with Python: analyzing text with the natural language toolkit.* " O'Reilly Media, Inc.", 2009.

[BMR20] T. B. Brown, B. Mann, N. Ryder, M. Subbiah, J. Kaplan, P. Dhariwal, et al. "Language models are few-shot learners." In *Advances in neural information processing systems*, pp. 1876–1901, 2020.

[BSC89] Ingo A Beinlich, Henri Jacques Suermondt, R Martin Chavez, and Gregory F Cooper. "The ALARM monitoring system: A case study with two probabilistic inference techniques for belief networks." In *AIME 89*, pp. 247–256. Springer, 1989.

[CBB19] Mathieu Chanson, Andreas Bogner, Dominik Bilgeri, Elgar Fleisch, and Felix Wortmann. "Privacy-preserving data certification in the Internet of things: Leveraging blockchain technology to protect sensor data." *Journal of the Association for Information Systems*, **20**(9), 2019.

[Cc23] Ricardo Cabello and Three.js contributors. "three.js - JavaScript 3D Library." `https://threejs.org/`, 2023. Accessed: 2023-02-21.

[CCW20] Steven Chen, Nicholas Carlini, and David Wagner. "Stateful detection of black-box adversarial attacks." In *Proceedings of the 1st ACM Workshop on Security and Privacy on Artificial Intelligence*, pp. 30–39, 2020.

[Cen23] National Deaf Center. "The ASL Interpreter Shortage and Its Impact on Accessibility in College Settings.", 2023. Accessed: September 11, 2023.

[Cep] Alpha Cephei. "VOSK Offline Speech Recognition API." `https://alphacephei.com/vosk/`. Accessed: 2022-11-29.

[Che21] Katelyn Cheng. "Deaf Schools Versus Mainstream Schools.", 2021. Accessed: 2023-03-25.

[CHI] MIKE CHIRICO. "Emergency - 911 Calls." `https://www.kaggle.com/datasets/mchirico/montcoalert`. Accessed: 2022-11-29.

[Chi21] Johnny Childers. "Google Chrome Extension Allows On-Screen ASL Signing For Disney+ Movies.", August 2021.

[Com23] Federal Communications Commission. "Text to 911: What You Need To Know.", 2023. Accessed: 2023-07-30.

[Con18] World Wide Web Consortium. "Web Content Accessibility Guidelines (WCAG) 2.1.", 2018.

[Cou14] Geoffrey R Coulter. *Current Issues in ASL Phonology: Phonetics and Phonology, Vol. 3*, volume 3. Academic Press, 2014.

[CSC17] Naomi K Caselli, Zed Sevcikova Sehyr, Ariel M Cohen-Goldberg, and Karen Emmorey. "ASL-LEX: A lexical database of American Sign Language." *Behavior research methods*, **49**:784–801, 2017.

[CT12] Thomas M Cover and Joy A Thomas. *Elements of information theory*. John Wiley & Sons, 2012.

[Dep] Eugene Police Department. "9-1-1 Call Scripts." `https://www.eugene-or.gov/2892/9-1-1-Call-Scripts`. Accessed: 2022-11-20.

[Dep20] Department of Justice. "ADA Requirements: Effective Communication.", 2020. Accessed: 2023-05-24.

[Dic21] Ellysse Dick. "Current and potential uses of AR/VR for equity and inclusion." Technical report, Information Technology and Innovation Foundation, 2021.

[DSA11] Rónán Daly, Qiang Shen, and Stuart Aitken. "Learning Bayesian networks: approaches and issues." *The knowledge engineering review*, **26**(2):99, 2011.

[ES03] George ElKoura and Karan Singh. "Handrix: animating the human hand." In *Proceedings of the 2003 ACM SIGGRAPH/Eurographics symposium on Computer animation*, pp. 110–119, 2003.

[FCZ17] Biyi Fang, Jillian Co, and Mi Zhang. "Deepasl: Enabling ubiquitous and non-intrusive word and sentence-level sign language translation." In *Proceedings of the 15th ACM conference on embedded network sensor systems*, pp. 1–13, 2017.

[FGK03] Sheila Frankel, R Glenn, and S Kelly. "The AES-CBC cipher algorithm and its use with IPsec." Technical report, 2003.

[Fri76] Lynn Alice Friedman. *Phonology of a soundless language: phonological structure of the American sign language.* University of California, Berkeley, 1976.

[GDL16] Ran Gilad-Bachrach, Nathan Dowlin, Kim Laine, Kristin Lauter, Michael Naehrig, and John Wernsing. "Cryptonets: Applying neural networks to encrypted data with high throughput and accuracy." In *ICML*, pp. 201–210, 2016.

[Gen09] Craig Gentry. "Fully homomorphic encryption using ideal lattices." In *STOC*, volume 9, pp. 169–178, 2009.

[GFZ19] Jiaying Gu, Fei Fu, and Qing Zhou. "Penalized estimation of directed acyclic graphs from discrete data." *Statistics and Computing*, **29**(1):161–176, 2019.

[GGM20] Sevgi Z Gurbuz, Ali Cafer Gurbuz, Evie A Malaia, Darrin J Griffin, Chris S Crawford, Mohammad Mahbubur Rahman, Emre Kurtoglu, Ridvan Aksu, Trevor Macks, and Robiulhossain Mdrafi. "American sign language recognition using rf sensing." *IEEE Sensors Journal*, **21**(3):3763–3775, 2020.

[Goo22] Google. "Filament." `https://github.com/google/filament`, 11 2022.

[GPM20] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. "Generative adversarial networks." *Communications of the ACM*, **63**(11):139–144, 2020.

[Gua13] The Guardian. "`https://www.theguardian.com/technology/appsblog/2013/sep/03/fitness-health-apps-sharing-data-insurance`.", 2013.

[Han04] Thomas Hanke. "HamNoSys-representing sign language data in language resources and language processing contexts." In *LREC*, volume 4, pp. 1–6, 2004.

[han23] "Hand talk: Your website accessible in ASL." `https://www.handtalk.me/en/`, Jan 2023.

[HLZ19] Jiahui Hou, Xiang-Yang Li, Peide Zhu, Zefan Wang, Yu Wang, Jianwei Qian, and Panlong Yang. "Signspeaker: A real-time, high-precision smartwatch-based sign language translator." In *The 25th Annual International Conference on Mobile Computing and Networking*, pp. 1–15, 2019.

[Hom23] "Homonyms in sign language." https://www.handspeak.com/learn/209, Mar. 2023.

[HT01] David J Hand and Robert J Till. "A simple generalisation of the area under the ROC curve for multiple class classification problems." *Machine learning*, **45**(2):171–186, 2001.

[HTF09] Trevor Hastie, Robert Tibshirani, Jerome H Friedman, and Jerome H Friedman. *The elements of statistical learning: data mining, inference, and prediction*, volume 2. Springer, 2009.

[HZX18] Tyler Hunt, Zhiting Zhu, Yuanzhong Xu, Simon Peter, and Emmett Witchel. "Ryoan: A distributed sandbox for untrusted computation on secret data." *ACM Transactions on Computer Systems (TOCS)*, **35**(4):1–32, 2018.

[Inc21] Ad Astra Inc. "Why ASL Interpreters are Hard to Find.", nov 2021. Accessed: 2023-05-23.

[INM23] "INMO Air." https://vr-compare.com/headset/inmoair, Mar. 2023.

[Ins23] "Insta 360 Go 2." `https://www.insta360.com/cn/product/insta360-go2/`, Mar. 2023.

[JCB20] Matthew Jagielski, Nicholas Carlini, David Berthelot, Alex Kurakin, and Nicolas Papernot. "High accuracy and high fidelity extraction of neural networks." In {*USENIX*} *Security*, pp. 1345–1362, 2020.

[JGZ21] Yincheng Jin, Yang Gao, Yanjun Zhu, Wei Wang, Jiyang Li, Seokmin Choi, Zhangyu Li, Jagmohan Chauhan, Anind K Dey, and Zhanpeng Jin. "SonicASL: An acoustic-based sign language gesture recognizer using earphones." *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, **5**(2):1–30, 2021.

[JJ12] Chen Jing and Fu Jingqi. "Fire alarm system based on multi-sensor bayes network." *Procedia Engineering*, **29**:2551–2555, 2012.

[JSM19] Mika Juuti, Sebastian Szyller, Samuel Marchal, and N Asokan. "PRADA: protecting against DNN model stealing attacks." In *2019 IEEE European Symposium on Security and Privacy (EuroS&P)*, pp. 512–527. IEEE, 2019.

[JSW21] Songyao Jiang, Bin Sun, Lichen Wang, Yue Bai, Kunpeng Li, and Yun Fu. "Skeleton aware multi-modal sign language recognition." In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 3413–3423, 2021.

[JVH19] Longlong Jing, Elahe Vahdani, Matt Huenerfauth, and Yingli Tian. "Recognizing american sign language manual signs from rgb-d videos." *arXiv preprint arXiv:1906.02851*, 2019.

[KN10] K. Korb and A. Nicholson. *Bayesian Artificial Intelligence*. Chapman and Hall, 2nd edition, 2010.

[Koe09] Philipp Koehn. *Statistical machine translation*. Cambridge University Press, 2009.

[Lad03] Paddy Ladd. *Understanding deaf culture: In search of deafhood*. Multilingual Matters, 2003.

[Lan05] Harlan Lane. "Ethnicity, ethics, and the deaf-world." *The Journal of Deaf Studies and Deaf Education*, **10**(3):291–310, 2005.

[LEM18] Taesung Lee, Benjamin Edwards, Ian Molloy, and Dong Su. "Defending Against Machine Learning Model Stealing Attacks Using Deceptive Perturbations." *arXiv preprint arXiv:1806.00054*, 2018.

[LJ89] Scott K Liddell and Robert E Johnson. "American sign language: The phonological base." *Sign language studies*, **64**(1):195–277, 1989.

[LRY20] Dongxu Li, Cristian Rodriguez, Xin Yu, and Hongdong Li. "Word-level deep sign language recognition from video: A new large-scale dataset and methods comparison." In *Proceedings of the IEEE/CVF winter conference on applications of computer vision*, pp. 1459–1469, 2020.

[LS88] Steffen L Lauritzen and David J Spiegelhalter. "Local computations with probabilities on graphical structures and their application to expert systems." *Journal of the Royal Statistical Society: Series B (Methodological)*, **50**(2):157–194, 1988.

[Luc01] Ceil Lucas. "The sociolinguistics of sign languages." 2001.

[M] M. "GitHub - MenacingDwarf/FacialTranslator." Accessed: 2023-02-21.

[Man] "Manifest.permission." `https://developer.android.com/reference/android/Manifest.permission`. Accessed: 2022-11-29.

[May07] Rachel I Mayberry. "When timing is everything: Age of first-language acquisition effects on second-language learning." *Applied psycholinguistics*, **28**(3):537–549, 2007.

[Med23] "MediaPipe Framework on Android." `https://developers.google.com/mediapipe/framework/getting_started/android`, Mar. 2023.

[MI94] Christine L MacKenzie and Thea Iberall. *The grasping hand*. Elsevier, 1994.

[MK02] Syed Atif Mehdi and Yasir Niaz Khan. "Sign language recognition using sensor gloves." In *Proceedings of the 9th International Conference on Neural Information Processing, 2002. ICONIP'02.*, volume 5, pp. 2204–2206. IEEE, 2002.

[MNS03] Leila Frances Monaghan, Karen Nakamura, Constanze Schmaling, and Graham H Turner. *Many ways to be deaf: International variation in deaf communities*. Gallaudet University Press, 2003.

[MS10] Yu Mao and Sean Shen. "Using Advanced Encryption Standard Counter Mode (AES-CTR) with the Internet Key Exchange version 02 (IKEv2) Protocol." 2010.

[MYB06] Ross E Mitchell, Travas A Young, Bellamie Bachelda, and Michael A Karchmer. "How many people use ASL in the United States? Why estimates need updating." *Sign Language Studies*, **6**(3):306–335, 2006.

[MZB21] Dominik Macháček, Matúš Žilinec, and Ondřej Bojar. "Lost in Interpreting: Speech Translation from Source or Interpreter?" *arXiv preprint arXiv:2106.09343*, 2021.

[Nat23] United Nations. "Sign languages unite us!", 2023. Accessed: 2023-05-25.

[NFO17] Asuka Noda, Osamu Fukuda, Hiroshi Okumura, and Kohei Arai. "Behavior analysis of a small animal using IoT sensor system." In *International Conference on Intelligent Informatics and Biomedical Sciences (ICIIBMS)*, pp. 9–10. IEEE, 2017.

[NKM00] Carol Neidle, Judy Kegl, Dawn MacLaughlin, Benjamin Bahan, and Robert G. Lee. *The Syntax of American Sign Language: Functional Categories and Hierarchical Structure*. MIT Press, 2000.

[NSL13] Radhakrishnan Nagarajan, Marco Scutari, and Sophie Lèbre. "Bayesian networks in R." *Springer*, **122**:125–127, 2013.

[OL11] Cemil Oz and Ming C Leu. "American sign language word recognition with a sensory glove using artificial neural networks." *Engineering Applications of Artificial Intelligence*, **24**(7):1204–1213, 2011.

[Ope20a] OpenAI. "71 PCT POLICE INVOLVED SHOOTING 911 TRANSCRIPT 4/4/2018." https://www.nyc.gov/assets/nypd/downloads/pdf/public_information/911-transcripts-police-involved-shooting-040418.pdf, 2020. Accessed: 2023-2-25.

[Ope20b] OpenAI. "Transcript of 911 call placed by Jason Ravnsborg on Saturday, September 12, 2020." https://dps.sd.gov/application/files/7216/0260/1522/911-call-transcribed.pdf, 2020. Accessed: 2023-2-25.

[Ope22] OpenAI. "chatGPT." https://chat.openai.com/chat, 2022. Accessed: 2023-2-29.

[Pad16] Carol A Padden. *Interaction of morphology and syntax in American Sign Language*. Routledge, 2016.

[Pai99] Pascal Paillier. "Public-key cryptosystems based on composite degree residuosity classes." In *International Conference on the Theory and Applications of Cryptographic Techniques*, pp. 223–238. Springer, 1999.

[Pea14] Judea Pearl. *Probabilistic reasoning in intelligent systems: networks of plausible inference*. Elsevier, 2014.

[Pen98] Xavier Pennec. *Computing the mean of geometric features application to the mean rotation*. PhD thesis, INRIA, 1998.

[PH88] C. Padden and T. Humphries. *Deaf in America: Voices from a Culture*. Harvard University Press, 1988.

[Pun16] Renée Punch. "Employment and adults who are deaf or hard of hearing: Current status and experiences of barriers, accommodations, and stress in the workplace." *American annals of the deaf*, **161**(3):384–397, 2016.

[PVG11] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. "Scikit-learn: Machine Learning in Python." *Journal of Machine Learning Research*, **12**:2825–2830, 2011.

[RBL22] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. "High-resolution image synthesis with latent diffusion models." In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 10684–10695, 2022.

[RKE21a] R. Rastgoo, K. Kiani, S. Escalera, and M. Sabokrou. "Sign language production: A Review." In *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, 2021.

[RKE21b] Razieh Rastgoo, Kourosh Kiani, and Sergio Escalera. "Sign language recognition: A deep survey." *Expert Systems with Applications*, **164**:113794, 2021.

[SBH17] Hossein Shafagh, Lukas Burkhalter, Anwar Hithnawi, and Simon Duquennoy. "Towards blockchain-based auditable storage and sharing of IoT data." In *Proceedings of the 2017 on Cloud Computing Security Workshop*, pp. 45–50, 2017.

[SBR20] Hossein Shafagh, Lukas Burkhalter, Sylvia Ratnasamy, and Anwar Hithnawi. "Droplet: Decentralized Authorization and Access Control for Encrypted Data Streams." In {*USENIX*} *Security*, pp. 2469–2486, 2020.

[SC07] Stan Salvador and Philip Chan. "Toward accurate dynamic time warping in linear time and space." *Intelligent Data Analysis*, **11**(5):561–580, 2007.

[SCB20] Ben Saunders, Necati Cihan Camgoz, and Richard Bowden. "Progressive transformers for end-to-end sign language production." In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XI 16*, pp. 687–705. Springer, 2020.

[SCC21a] Zed Sevcikova Sehyr, Naomi Caselli, Ariel M Cohen-Goldberg, and Karen Emmorey. "The ASL-LEX 2.0 Project: A database of lexical and phonological properties for 2,723 signs in American Sign Language." *The Journal of Deaf Studies and Deaf Education*, **26**(2):263–277, 2021.

[SCC21b] Zed Sevcikova Sehyr, Naomi Caselli, Ariel M Cohen-Goldberg, and Karen Emmorey. "The ASL-LEX 2.0 Project: A database of lexical and phonological properties for 2,723 signs in American Sign Language." *The Journal of Deaf Studies and Deaf Education*, **26**(2):263–277, 2021.

[SCH20] Stephanie Stoll, Necati Cihan Camgoz, Simon Hadfield, and Richard Bowden. "Text2Sign: towards sign language production using neural machine translation and generative adversarial networks." *International Journal of Computer Vision*, **128**(4):891–908, 2020.

[Sch21] Philippe Schlenker. "Iconic presuppositions." *Natural Language & Linguistic Theory*, **39**:215–289, 2021.

[Scu09] Marco Scutari. "Learning Bayesian networks with the bnlearn R package." *arXiv preprint arXiv:0908.3817*, 2009.

[SD74] Jerome D Schein and Marcus T Delk Jr. "The deaf population of the United States." 1974.

[SDK18] Bowen Shi, Aurora Martinez Del Rio, Jonathan Keane, Jonathan Michaux, Diane Brentari, Greg Shakhnarovich, and Karen Livescu. "American sign language fingerspelling recognition in the wild." In *2018 IEEE Spoken Language Technology Workshop (SLT)*, pp. 145–152. IEEE, 2018.

[Sha21] Ethan Shanfeld. "New Google Chrome Extension SignUp Offers ASL Captions for Three Films on Disney Plus.", August 2021.

[SHB17] Hossein Shafagh, Anwar Hithnawi, Lukas Burkhalter, Pascal Fischli, and Simon Duquennoy. "Secure Sharing of Partially Homomorphic Encrypted IoT Data." In *Proceedings of the 15th ACM Conference on Embedded Network Sensor Systems*, p. 29. ACM, 2017.

[SHB20] Stephanie Stoll, Simon Hadfield, and Richard Bowden. "Signsynth: Data-driven sign language video generation." In *Computer Vision–ECCV 2020 Workshops: Glasgow, UK, August 23–28, 2020, Proceedings, Part IV 16*, pp. 353–370. Springer, 2020.

[SHD15] Hossein Shafagh, Anwar Hithnawi, Andreas Dröscher, Simon Duquennoy, and Wen Hu. "Talos: Encrypted query processing for the internet of things." In *Sensys*, pp. 197–210. ACM, 2015.

[Sig23a] "Sign Language Translator." https://apps.apple.com/us/app/sign-language-translator/id1458992650, Mar. 2023.

[Sig23b] "Signing Savvy." `https://www.signingsavvy.com/`, Mar. 2023.

[SM02] Richard J Senghas and Leila Monaghan. "Signs of their times: Deaf communities and the culture of language." *Annual Review of Anthropology*, **31**(1):69–97, 2002.

[Son23] Filip Sondej. "Autocorrect.", Mar. 2023.

[SPP05] Karen Sachs, Omar Perez, Dana Pe'er, Douglas A Lauffenburger, and Garry P Nolan. "Causal protein-signaling networks derived from multiparameter single-cell data." *Science*, **308**(5721):523–529, 2005.

[SRT10] Lucia Specia, Dhwaj Raj, and Marco Turchi. "Machine translation evaluation versus quality estimation." *Machine translation*, **24**:39–50, 2010.

[SSB20] Qijia Shao, Amy Sniffen, Julien Blanchet, Megan E Hillis, Xinyu Shi, Themistoklis K Haris, Jason Liu, Jason Lamberton, Melissa Malzkuhn, Lorna C Quandt, et al. "Teaching american sign language in mixed reality." *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, **4**(4):1–27, 2020.

[STO60] WC STOKOE. "Sign Language Structure: An Outline of the Visual Communication System of the American Deaf." *Studies in Linguistics: Occasional Papers*, **8**, 1960.

[TGB10] Richard A Tennant, Marianne Gluszak, and Marianne Gluszak Brown. *The American sign language handshape dictionary*. Gallaudet University Press, 2010.

[The22] "The ASL Interpreter Shortage and Its Impact on Accessibility in College Settings." https://nationaldeafcenter.org/news-items/the-asl-interpreter-shortage-and-its-impact-on-accessibility-in-college-settin Dec. 2022.

[Tra23] "Transmission Rate vs. Bandwidth in Bluetooth Technology." https://resources.pcb.cadence.com/blog/2022-transmission-rate-vs-bandwidth-in-bluetooth-technology, Mar. 2023.

[Uni23] European Union. "European Accessibility Act.", 2023.

[Ver19] Verizon. "https://enterprise.verizon.com/resources/reports/dbir/.", 2019.

[Vic23] Bill Vicars. "Classifiers.", 2023. Accessed: 2023-07-15.

[Viv23] "Vivo Sign Language Translator." https://assist.vivo.com/detail/translate, Mar. 2023.

[VL00] Clayton Valli and Ceil Lucas. *Linguistics of American sign language: An introduction*. Gallaudet University Press, 2000.

[VOI23] "VOISS." https://www.projectvoiss.org/, Mar. 2023.

[VSP17] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. "Attention is all you need." *Advances in neural information processing systems*, **30**, 2017.

[Vuz23] "Vuzix Blade 2." https://vr-compare.com/headset/vuzixblade2, Mar. 2023.

[Wik22] Wikipedia contributors. "Mixamo — Wikipedia, The Free Encyclopedia.", 2022. [Online; accessed 7-December-2022].

[WJC19] Yue Wang, Ziyu Jiang, Xiaohan Chen, Pengfei Xu, Yang Zhao, Yingyan Lin, and Zhangyang Wang. "E2-Train: Training State-of-the-art CNNs with Over 80% Energy Savings." In *Advances in Neural Information Processing Systems*, pp. 5139–5151, 2019.

[ZA23] Lvmin Zhang and Maneesh Agrawala. "Adding conditional control to text-to-image diffusion models." *arXiv preprint arXiv:2302.05543*, 2023.

[ZBB18] Yuhang Zhao, Cynthia L Bennett, Hrvoje Benko, Edward Cutrell, Christian Holz, Meredith Ringel Morris, and Mike Sinclair. "Enabling people with visual impairments to navigate virtual reality with a haptic and auditory cane simulation." In *Proceedings of the 2018 CHI conference on human factors in computing systems*, pp. 1–14, 2018.

[ZBV20] Fan Zhang, Valentin Bazarevsky, Andrey Vakunov, Andrei Tkachenka, George Sung, Chuo-Ling Chang, and Matthias Grundmann. "Mediapipe hands: On-device real-time hand tracking." *arXiv preprint arXiv:2006.10214*, 2020.

[ZCL20] Zhihao Zhou, Kyle Chen, Xiaoshi Li, Songlin Zhang, Yufen Wu, Yihao Zhou, Keyu Meng, Chenchen Sun, Qiang He, Wenjing Fan, et al. "Sign-to-speech translation using machine-learning-assisted stretchable sensor arrays." *Nature Electronics*, **3**(9):571–578, 2020.

[ZJW22] Qian Zhang, JiaZhen Jing, Dong Wang, and Run Zhao. "Wearsign: Pushing the limit of sign language translation using inertial and EMG wearables." *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, **6**(1):1–27, 2022.

[ZP15] Wei Zhou and Selwyn Piramuthu. "IoT and supply chain traceability." In *International Conference on Future Network Systems and Security*, pp. 156–165. Springer, 2015.