Real-Time Bangla Sign Language Translation: Bidirectional Voice-to-3D Avatar and Sign-to-Voice Conversion

SHARIFUR RAHMAN

MASTER OF SCIENCE IN INFORMATION AND COMMUNICATION ENGINEERING

NOAKHALI SCIENCE AND TECHNOLOGY UNIVERSITY

Real-Time Bangla Sign Language Translation: Bidirectional Voice-to-3D Avatar and Sign-to-Voice Conversion

SHARIFUR RAHMAN

This report was submitted in fulfillment of the requirements for the award of the degree of Master of Science in Information and Communication Engineering

Department of Information and Communication Engineering

NOAKHALI SCIENCE AND TECHNOLOGY UNIVERSITY

SEPTEMBER 2025

STUDENT'S DECLARATION

This research proposal is submitted to the Department of Information and Communication Engineering (ICE), Noakhali Science and Technology University, Noakhali- 3814, in partial fulfillment of the requirements for the M.Sc. degree in ICE under the course entitled "ICE- 5321". So, I, here by declare that this thesis proposal has not been submitted elsewhere for the requirement of any kind of degree, diploma or publication.

Sharifur Rahman

Roll No: ASH2211MSc115M

Session: 2021-22

Year: 2, Term: I

Department of Information and Communication Engineering

Noakhali Science and Technology University
Noakhali- 3814.

SUPERVISOR'S DECLARATION

We hereby declare that we have checked this thesis and in our opinion, this thesis is adequate in terms of scope and quality for the award of the degree of Master of Science in Information and Communication Engineering.

_____

Name of Supervisor: DR. MD. ASHIKUR RAHMAN KHAN
Position: PROFESSOR & CHAIRMAN
Date:

_____

Name of Co-Supervisor: DR. MOHAMMAD AMZAD HOSSAIN
Position: ASSOCIATE PROFESSOR
Date:

ABSTRACT

In this paper, we have presented a real-time, bidirectional communication system for translating spoken Bangla to Bangla Sign Language (BdSL) and sign to text and voice translation. We have used 3D avatar-based animations. The main objective is to enhance communication between hearing and hearing-impaired people in Bangladesh. In Bangladesh, a significant portion of the population suffers from hearing impairments. The Voice-to-Sign Translation system captures spoken Bangla sentences using a microphone. Then it was transcribed to text via the Google Cloud Speech API. The text was then sent for tokenization, parsing, and lemmatization. And finally, the individual word is mapped to BdSL gestures. These gestures are represented using HamNoSys notation. SiGML files are used to animate 3D avatars performing the signs. In the Sign-to-Voice Conversion system, we have used a Transformer-based classification model to recognize BdSL gestures. This offers significant advantages over traditional CNNs and RNNs by processing sequential gesture data in parallel. Their ability to process sequences in parallel and capture long-range dependencies efficiently leads to lower computational overhead and faster processing speeds. This feature increases speed in both training and inference. The use of MediaPipe for keypoint extraction further streamlines the process by reducing the complexity of raw video data and allowing the model to focus on meaningful gesture features, making the entire system more efficient and accurate. The voice-to-sign system was evaluated in four categories: alphabet, numbers, words, and sentences. We have evaluated it by seven professional sign language experts from two different institutes. Results for the alphabet showed an average accuracy of 92.5%. For numbers, the system achieved an average accuracy of 98.20%. In the words category, the system achieved an accuracy of 82.94%. For sentences, it has achieved an average accuracy of 82.20%. The Sign-to-Voice Conversion system achieved a validation accuracy of 83.2% with a Macro F1-score of 0.7806. This gives a good overall performance. Misclassification was indicated by some of the signs because of visual ambiguities. There was also real-time processing with a mean CPU time per sign word of 1.58 seconds. This study paper is evidence of the possibilities in uniting natural language processors with deep learning and 3D avatar animation. It may be applied to develop a scaled-up interactive communication platform that is efficient and real-time. The findings justify the utility of the system in the translation of spoken Bengali into BdSL and BdSL to spoken Bengali.

Keywords: Real-Time Communication, Sign-to-Voice Conversion, 3D Avatar, Transformer CLS, Bangla Sign Language (BdSL), Voice-to-Sign Translation

TABLE OF CONTENTS

CHAPTER 4 RESULTS AND DISCUSSION

CHAPTER 5 CONCLUSION

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

1.1 INTRODUCTION

This chapter gives a prefatory discussion of our research work. The contribution of this research is mainly focused on fabricating a real-time bidirectional communication system that translates Bangla speech into Bangla Sign Language (BdSL) via 3D avatars and BdSL gestures revert into Bangla text and speech. Thus, in the chapter, we start with a background study on the communication challenges of the Bangladeshi deaf community and the need for accessible technological solutions. Following this, the problem statement is presented to highlight the gaps in existing systems and the necessity for a bidirectional, real-time solution for the Bangla language. Based on the encountered problems, the research objectives are then defined, outlining the specific goals of building robust voice-to-sign and sign-to-voice modules, using 3D avatar technology, and enabling real-time translation accuracy. The scope of the study is subsequently presented, broadcasting how this research extends previous works on static or isolated sign recognition by introducing a dynamic, scalable, and effective system for continuous real-world communication. Finally, the outline of the thesis is provided at the end of the chapter, offering an overview of the structure and flow of the work.

1.2 BACKGROUND

Communication is a prime human need. Sign language serves as the foremost mode of interaction for the hearing-impaired community. Nevertheless, major obstacles to effective communication are created by the lack of widespread sign language proficiency among the general population. In Bangladesh, one of the major health issues is deafness. According to the 2011 census, around 9.6% of the country's population, approximately 13.7 million people, experience deafness or hearing difficulties, defined as a hearing loss of 40 dB or more. In addition, intense hearing loss, described by a hearing impairment of 90 DB or more, affects 1.2% of the population, equivalent to about 1.7 million individuals. Bangla Sign Language (BdSL) is the sign language used in Bangladesh. Deaf people in Bangladesh often lack access to treatment or the right educational methods. They also commonly face discrimination due to their disability. To improve the lives of hearing-impaired individuals, *numerous* Deaf associations in Bangladesh are now working [1].

In this paper, a Dual-Mode Communication System is proposed to accelerate real-time bidirectional translation between Bangla voice and Bangla Sign Language (BdSL) and vice versa. Two integrated modules are adopted in the system: First, it converts Bangla voice into dynamic 3D avatar animations that represent BdSL gestures. Then the second one converts BdSL gestures back into Bangla text and voice. This all-inclusive approach ensures smooth communications between hearing and hearing-impaired people. This approach also aims to promote social inclusion and improve access to information. State-of-the-art natural language processing (NLP) techniques and HamNoSys notation are both applied in the voice-to-sign for an accurate translation of spoken Bangla to equivalent BdSL signs. SiGML files are used to create 3D avatars to bring the movements to life. Still, the sign-to-voice/text module applies computer vision techniques (convolutional neural networks (CNNs) and K-Nearest Neighbors (KNN))to identify and interpret BdSL gestures. The identified gestures are then turned into Bangla text and synthesized into normal Bangla speech via text-to-speech (TTS) technology[2]. In this research, we expect to upgrade communication accessibility for people with hearing impairments by combining both modules into one system. This way, our dual-mode system promotes seamless communications, besides aligning with the overall picture of digital inclusivity and change in Bangladesh. The World Health Organization writes that, over 430 million people worldwide are deaf nowadays, and this figure is bound to increase to 700 million by the year 2050 (WHO, 2021)[1]. This presents the global community with a matter of urgency to find universally accessible communication technologies. The 2011 national census reports that approximately 9.6% of the global population (≈ 13.7 million people) live with significant hearing loss (≥ 40 dB). Moreover, 34.6% of the global population(≈ 49.2 million people) have

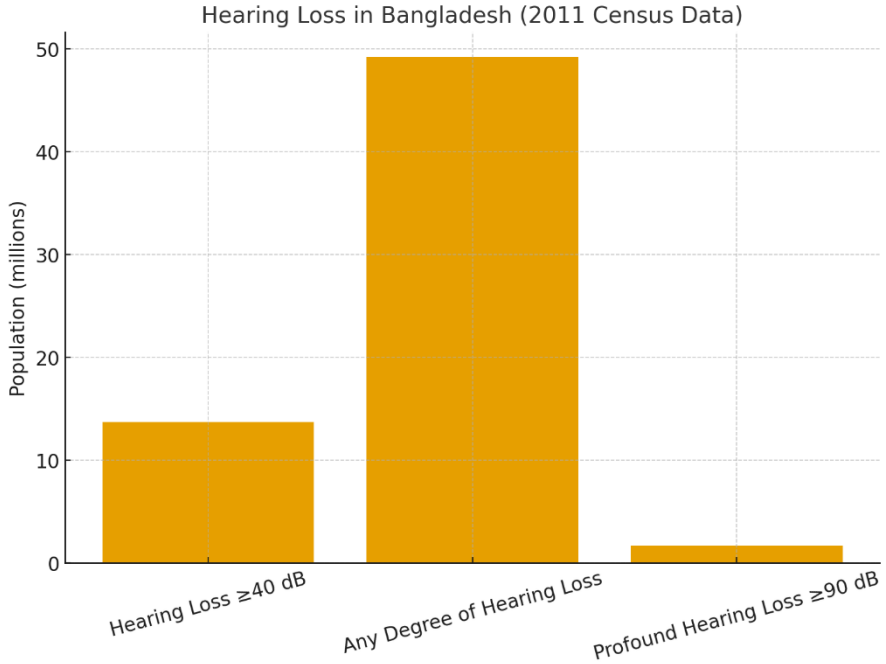


Figure 1.1: Prevalence of hearing loss in Bangladesh.

some degree of hearing loss, and 1.2% of the global population (≈ 1.7 million people) experience profound hearing loss above 90 dB. [2]. Despite these figures, awareness and accessibility remain limited. BdSL is not widely taught in mainstream schools, interpreter

services are scarce, and many deaf individuals face discrimination or exclusion from education, healthcare, and employment. As a result, communication breakdowns are common, particularly in hospitals, legal institutions, and public services. These barriers underscore the importance of technological interventions, especially automated sign language translation systems, to foster inclusion and independence for the deaf community.

Evolution of Sign Language Recognition Systems Worldwide

Research on automatic sign language recognition (SLR) and translation has developed considerably since the 1990s, largely driven by advances in computer vision, natural language processing, and machine learning. These works can be broadly divided into:

1. Vision-based approaches, cameras capture hand and body movements, which are analyzed using image processing or deep learning.
2. Sensor-based approaches gloves or wearables, detect hand flexion and motion, converting them into digital signals.
3. Hybrid and multimodal systems combining cameras, sensors, and linguistic models to improve recognition.

Early Systems

One of the pioneering projects, Tessa (2002) in the UK, introduced a speech-to-sign avatar to support interactions in postal services. It converted spoken English into British Sign Language (BSL) gestures, animated by a virtual character. While innovative, its limited vocabulary and unrealistic animation restricted usability [1].

In the U.S., significant work focused on American Sign Language (ASL). A milestone was DeepASL (2018), which used infrared sensors and deep learning to recognize word- and sentence-level signs with up to 94.5% accuracy. More recent systems employ Transformer-based architectures, achieving higher accuracy by modeling temporal dependencies across sign sequences. [2].

Advances in Other Asian Languages

- **Hindi Sign Language (HSL):** CNN + LSTM models have achieved ~95% accuracy in recognizing both isolated and continuous gestures [3]
- **Arabic Sign Language (ArSL):** Hybrid methods combining **glove sensors** with **camera-based tracking** have reported ~90% recognition accuracy [4].
- **Chinese Sign Language (CSL):** Large-scale datasets and deep neural networks have enabled recognition of thousands of words with over 96% accuracy [5] .

These efforts show a global trend toward **deep learning and multimodal integration**. However, most works remain **language-specific**, making them difficult to adapt directly to BdSL due to differences in grammar, morphology, and hand-shape lexicons.

Table 1.1: Comparative Study of Sign Language Translation Systems

Language	Approach	Accuracy / Performance

ASL (American)	DeepASL: infrared sensors + deep learning (2018) [6]	Word-level acc. 94.5%, sentence-level support
BSL (British)	Tessa Avatar: speech-to-sign avatar (2002) [1]	Limited vocabulary; simple animations
Hindi SL	CNN + LSTM for isolated & continuous signs (2019) [7]	Accuracy ~95% on benchmark dataset
Arabic SL	Glove-based + vision hybrid system (2020) [8]	Accuracy ~90% depending on the dataset
Bangla SL	Borno-Net YOLOv4-Tiny + LSTM; BdSL47 ANN (2023) [9]	mAP 99.7% (Borno-Net); F1 = 97.8% (BdSL47)

Avatar-Based Sign Synthesis

While recognition systems translate signs into text or speech, avatar-based systems attempt the reverse: translating text or speech into sign language gestures. Avatars provide dynamic, visual representations of signs, often animated in real time. Early avatar systems were limited by jerky movements, a lack of non-manual features (facial expressions, lip movements), and grammatical mismatches. However, frameworks such as HamNoSys (Hamburg Notation System) and SiGML (Signing Gesture Markup Language) significantly improved sign synthesis [10]. HamNoSys provides a symbolic notation for gestures (hand shape, movement, orientation), while SiGML translates these notations into animation commands for 3D avatars [11].

Recent studies highlight the importance of linguistic accuracy and naturalistic animation for acceptance among deaf users. Lau et al. (2023) reviewed 40 years of avatar systems, emphasizing that while many projects succeeded in technical implementation, they failed to meet community expectations due to unnatural signing. Commercial efforts such as Silence Speaks (2024) now use AI-driven avatars to translate text into BSL/ASL, showing promising scalability. Still, no comparable avatar-based systems exist for Bangla, marking a critical gap. [12].

Recent Advances in Bangla Sign Language (BdSL) Recognition: Here, we have discussed the recent works that have been done on Bangla Sign Language.

- Sensor-Based Systems: Early BdSL research explored flex-sensor gloves connected to Arduino microcontrollers, translating specific finger bends into corresponding BdSL signs. While useful for controlled experiments, these devices are impractical for daily communication due to cost, portability issues, and the unnatural burden of wearing sensors [13] .

- Image Processing Approaches: Researchers have employed Support Vector Machines (SVM), k-Nearest Neighbors (KNN), and Bag-of-Features models to recognize BdSL alphabets and numbers from images. Accuracy levels ranged from 70–86%, but these methods were limited by sensitivity to lighting and backgrounds [14].

- Skeletal Tracking: The Microsoft Kinect sensor was employed to capture body skeletons and recognize simple BdSL gestures. While effective for structured datasets, the Kinect hardware is not widely available in Bangladesh, limiting its scalability [15] .

- Deep Learning Method: More recent approaches utilize large datasets and deep neural networks:

    i. **Borno-Net (2023):** A YOLOv4-Tiny + LSTM model trained on the Okkhornama dataset, capable of recognizing 49 classes with a mean Average Precision (mAP) of 99.7% and sentence-level accuracy of 99.1% [16].
    ii. **BdSL47 Dataset (2023):** Introduced 47 depth-based sign classes, with Artificial Neural Network (ANN) classifiers achieving an F1 score of 97.8% [17] .
    iii. **Hybrid Approaches:** Real-time BdSL detection deployed on Jetson Nano devices, enabling portable, low-power solutions [18].
    iv. **BAUST Lipi Dataset (2024):** Introduced a new BdSL dataset comprising alphabets totaling 18,000 images, with each image being 224x224 pixels in size. The dataset encompasses 36 Bengali symbols, of which 30 are consonants and the remaining six are vowels. A hybrid Convolutional Neural Network (CNN) model, integrating multiple convolutional layers, activation functions, dropout techniques, and LSTM layers, achieved an accuracy rate of 97.92% upon evaluation with this dataset [19].
    v. **BdSLW60 Dataset (2024):** A comprehensive BdSL word-level dataset named BdSLW60 was created, encompassing 60 Bangla sign words with a significant scale of 9307 video trials provided by 18 signers under the supervision of a sign language professional. The dataset was rigorously annotated and cross-checked by 60 annotators. An attention-based bi-LSTM model achieved testing accuracy up to 75.1% on this dataset [20].
    vi. **Finger Spelling Recognition (2023):** A YOLOv5-based real-time finger spelling system for Bangla Sign Language, named BdSpell, was developed. The system employs specified rules and numerical classes as triggers to efficiently generate hidden and compound characters, eliminating the necessity for additional classes. It achieves character spelling in an impressive 1.32 seconds with a remarkable accuracy rate of 98% [21].
    vii. **Video Transformer Models (2025):** Fine-tuning state-of-the-art video transformer architectures, VideoMAE, ViViT, and TimeSformer—on BdSLW60 and BdSLW401 datasets demonstrated that video transformer models significantly outperform

traditional machine learning and deep learning approaches. Among the models, the VideoMAE variant achieved the highest accuracy of 95.5% on the frame rate corrected BdSLW60 dataset and 81.04% on the front-facing signs of BdSLW401 [22].

Key Gaps and Research Opportunities

Despite global and national advancements, several critical gaps remain:

i. **Lack of bidirectional systems:** Current BdSL projects focus mainly on recognition (sign → text/speech).
ii. **Absence of real-time integration:** Most systems work offline or in controlled datasets but fail in dynamic, real-world environments.
iii. **Insufficient user validation** – Few studies involve deaf interpreters or BdSL-native users for evaluation.

1.3 PROBLEM STATEMENT

In Bangladesh, approximately 9.6% of the population, equating to about 13.7 million people, are affected by hearing impairment, with a significant portion being children under 18 years old. This substantial demographic faces profound challenges in communication, education, and social integration due to the limited development of Bangla Sign Language (BdSL) and the absence of effective assistive technologies [23].

BdSL is still at a developing stage with respect to standardized vocabulary in particular. Research indicates that current BdSL dictionaries have fewer than 5,000 words and gestures, which is far less than the large vocabulary for effective daily communications. The lack of vocabulary makes it extremely difficult for deaf people to express abstract concepts, emotions, or technical terms. These terms restrict their participation in education as well as in professional fields. The limitations can slow down cognitive growth, literacy development, and socialization for children. They often cannot communicate like normal kids with their teachers and classmates. Deaf children in Bangladesh face many barriers to formal schooling, including not enough skilled instructors in BdSL and curricula that are not tailored for sign language students. Most of the time, they rely on relatives or volunteer interpreters. But that's not enough. This results in a lack of classroom participation or engagement. The need for real-time voice-to-sign translation further limits the technological reach of digital educational materials. Literacy among deaf children lags far behind the national average literacy rate. These limitations reinforce educational disparities. In the education sector, limited communication skills frequently result in a social barrier for many deaf people. Being excluded from normal conversations, public events, and workplace discussions. Without the normal conversation, they often foster feelings of marginalization, depression, and anxiety. Deaf children struggle to build friendships. The adults often encounter difficulties in securing stable employment or navigating essential services independently. In families constructed with the role of informal interpreters, this limits personal independence and overall productivity. Bangladesh's digital infrastructure has progressed so much that the technical solutions for BDSL have been preliminary. Current systems focus on systematic conversion using static videos or recognizing single gestures only. These solutions are not enough for real-time communication and cannot support complex communication. Besides, solutions for other sign languages cannot be directly transferred to Bangla due to its unique linguistic features. The unique features of the Bengali language include speech formation, articulation, and context-dependent word usage. They significantly hinder the easy adaptation of the existing word search.

The lack of voice-to-sign and sign-to-voice two-way systems that can convert themselves makes it more

difficult for the hearing impaired to communicate. Inclusive environments are crucial for individuals, where the deaf can independently participate in social, educational, and developmental areas. Creating such a system will not only ensure social participation but also ensure social control. Especially in the case of the BdSL vocabulary, there is a huge gap. To foster inclusive environments, bridging this gap is critical, where deaf individuals can engage independently in social, educational, and professional spheres. Creating such a system will strengthen social participation as well as guarantee fair access to essential services across society.

Table 1.2: Key Challenges Faced by the Bangla-Speaking Deaf Community.

Category	Problem Description	Impact on Deaf Individuals
Educational Barriers[24]	Insufficient number of trained instructors and sign language–adapted learning materials.	Reduced literacy rates, partial lesson comprehension, and reliance on family interpreters.
Social Isolation[25]	Communication difficulties in public spaces, schools, workplaces, and social events.	Feelings of marginalization, anxiety, depression; difficulties in forming relationships; reduced participation in community life.
Technological Gaps[26]	Existing BdSL tools are mostly text-to-sign or pre-recorded videos; no real-time voice-to-sign or bidirectional systems.	Limits interactive communication; inadequate for live classrooms, broadcasting, and government services.
Professional Support Scarcity	Few certified interpreters are available in urban and rural areas.	Dependence on family/friends; reduced personal autonomy; limited access to healthcare, legal, and public services.
Adaptation Issues from Other SL Systems	Systems for Hindi, ASL, or English cannot be directly applied due to Bangla grammar, syntax, and context-specific word usage.	Current technologies fail to meet the specific needs of the Bangla-speaking deaf population.

1.4 OBJECTIVES

The overarching objective of this research project is to design and develop an automated, bidirectional communication system. This system will bridge the linguistic gap between the hearing and deaf communities in Bangladesh by enabling seamless translation between Bangla speech and Bangla Sign Language (BdSL). Our research aims to create a real-time Bangla voice-to-sign language translation system capable of accurately processing natural Bangla speech inputs and converting them into dynamic BdSL gestures, ensuring that both simple words and complex sentence structures are correctly represented. At the same time, a robust gesture recognition module will be implemented to detect and interpret BdSL gestures, subsequently translating them into Bangla text and voice outputs in real time, thereby enabling inclusive two-way interaction. The proposed system is intended to provide a natural and expressive process of communication through a dynamic 3D avatar animation. BdSL gestures will

be generated through HamNoSys notation in combination with SiGML scripting. This method provides a precise representation of signs and increases visual understanding. This approach ensures precise articulation of signs and improves visual clarity. The research will emphasize adaptability and scalability, which enable the system to accommodate a wide range of communication scenarios, including numbers, words, and complete sentences—areas where current systems often fall short. Further, there will be intensive testing and validation carried out with the involvement of professional BDSL interpreters to evaluate the accuracy of the system, linguistic appropriacy, as well as the overall realism of the acting by the avatar, so the system is used for actual purposes of communication. Finally, to maximize usability, there will be the integration of a convenient interface for facilitating smooth interaction across different domains. These include education, where deaf students struggle with hearing verbal lectures; healthcare facilities, where proper communication between physicians and patients is essential; media and broadcasting, where accessible transmission of information is imperative; and public services, where equitable communication resources are necessary. Our project will favor preferences in the field of sign language recognition and synthesis, and simultaneously, enable the deaf community in Bangladesh to be socially inclusive and empowered.

This was a bidirectional communication system that would translate the Bangla speech into Bangla Sign Language (BDSL) and vice versa, which would translate BDSL gestures to Bangla text and voice. This again contributed towards the originality of this study. In general, the real objectives of this thesis work are the following:

1. Design a dynamic 3D avatar animation system that generates and renders BDSL gestures in HamNoSys notation and SiGML scripting.
2. Ensures that the system is scalable and adaptable to handle different communication scenarios (numbers, words, and long sentences).
3. Validate and verify the system's accuracy and usability through feedback from experienced Bangla Sign Language interpreters in relation to linguistic accuracy and avatar performance.
4. Create a user-friendly interface that facilitates uninterrupted communication for various real-world applications, such as educational tools, healthcare communication, broadcast media, and public service platforms.
5. Develop and implement a robust gesture recognition module that can transform BDSL gestures into Bangla text and voice in real time.

1.5 CONTRIBUTION

The contributions of this research to the field of BdSL translation and communication are significant. We have addressed key gaps and challenges in the existing systems. The main contributions of the research are as follows:

Bidirectional Real-Time Communication System: The research introduces a novel bidirectional system. This can decode spoken Bangla language with 3D avatar styles and the other way round. It has the ability to translate BdSL gestures into voice and Bangla text. This system provides successful communication between hearing and hearing-impaired people in real time.

Transformer-Based Gesture Recognition for Sign-to-Voice Conversion: Gesture recognition is done using a Transformer-based model. This allows the system to interpret BdSL gestures and translate it to Bangali text or speech. The given process has an advantage over conventional CNNs and RNNs as it enables the processing of sequential data in parallel, enhancing the speed and efficiency. This method is useful, particularly when used in real-time.

Dataset Creation and Diversity: The study adds to the establishment of an effective database that contains a variety of signers, settings, and backgrounds. This dataset consists of 1,869 videos containing 135 different BdSL gestures, and it is essential in the process of training the gesture recognition system. The age, gender, and signing style diversity among the signers is beneficial in making the model

applicable to a real-world environment, since they can solve the problems of imbalance of the dataset and the variation of the environment. We have also developed 800 frequently used Bangali words, letters, and digits that could be utilized in learning, practicing, and real-time conversion of language into voice-to-sign.

To conclude, the study has filled the technological gap in the BdSL communication systems by providing a real-time, two-way translation system that combines modern machine learning mechanisms and the 3D avatars animation. It offers a scaling solution to support inclusiveness and better communication to the deaf and the hearing-impaired community.

1.6 SCOPE OF THE STUDY

Background studies on Bengali Sign Language (BDSL) translation have focused on the formation of letters, or simple words, using real-time still images or pre-recorded gestures. Even though this provides a guideline, they have not been able to translate Bengali words in real-time, which is a significant step towards practical applications for communication. In our research, we propose to fill this gap by developing a comprehensive, two-way communication system that can translate Bengali speech and Bengali Sign Language (BDSL) in real-time. While supporting discrete gestures, the proposed system is designed to support continuous natural conversation between hearing and hearing-impaired individuals. The system will consider language relevance and advanced machine learning to learn and translate complete Bengali language words and speech from previous real-time BDSL gestures. In the background, the augmented reality 3D avatar will perform fluid and precise sign gestures in real-time, with real-time voice input. This environment will be tested in a variety of environments - educational environments, healthcare, public relations, and broadcast - to demonstrate both scalability and adaptability. Based on the value of the public and user feedback, the project seeks to provide a user-friendly solution that improves communication and increases the inclusion of deaf people in society.

1.7 ORGANIZATION OF THE THESIS

We have organized our research report writing by following the given pathway:

Chapter 1: Introduction: Establishes the motivation for real-time Bangla Sign Language (BSL) communication, the problem statement, objectives, and scope. It explains the bidirectional goal converting speech/text to a sign avatar and converting sign videos to text/voice. It also covers research questions, significance for accessibility, and a brief outline of the thesis structure.

Chapter 2: Literature Review: Surveys prior work on sign language recognition, voice to sign pipelines (ASR, lexical mapping, avatar animation), and sign to text/voice systems. It compares datasets and benchmarks, identifies gaps for Bangla resources, and motivates the chosen design choices.

Chapter 3: Methodology: Creation of SigML files and development (collection, cleaning, splits) of datasets, preprocessing (keypoint extraction), etc. They are described in terms of training arrangements, testing procedures, and details of implementation to be reproducible.

Chapter 4: Results & Discussion: Training curves, validation, confusion, and per-class. It will textualize strengths, limitations, and connection findings to previous work.

Chapter 5: Conclusion: Summarizes contributions across both highlights practical implications for inclusive communications.

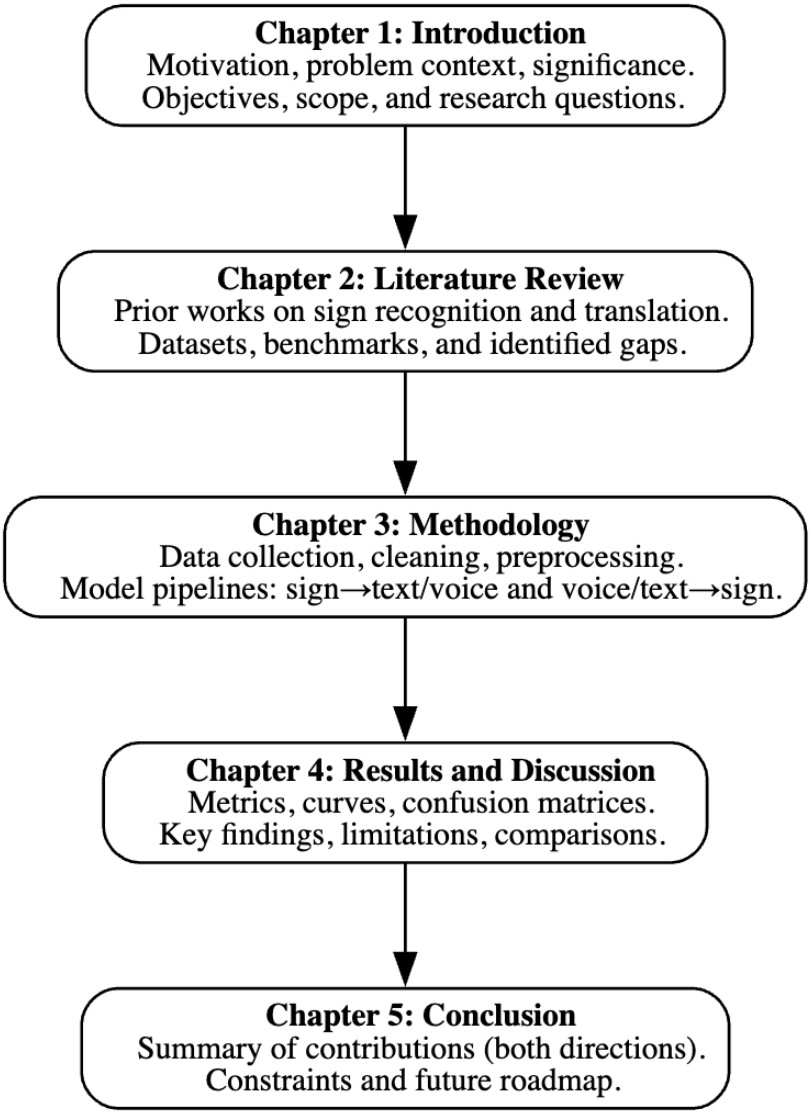


Figure 1.2: Organization of our research work.

CHAPTER 2

LITERATURE REVIEW

1.1 INTRODUCTION

The research chapter will offer an in-depth autonomous examination of the amount of research on translation systems based on Bangla Sign Language (BdSL). Our literature review relates to current projects with respect to recent research on sign languages. With this literature review, we aim to study the methods used for both voice-to-sign and sign-to-voice translators. Sign language technology is developed by considering deep learning models for gesture detection and through the use of a 3D avatar animation. We are referring to the existing system and paying particular attention to the real-time communication translation that is done within Bangladesh, as we specifically deal with Bangla Sign Language in this research. This literature review identifies findings from previous studies, outlines the effectiveness of various approaches, and provides directions for the current research by outlining a plan for an improved, real-time solution for the Bengali-speaking deaf community.

1.2 VOICE TO SIGN CONVERSION

With a spotlight on languages in various cultural contexts, recent research on automated sign language systems is thriving worldwide. However, Bangla Sign Language(BdSL) is spoken by millions of individuals with hearing impairments in Bangladesh as well as in some surrounding regions where still lack viable technological solutions compared to other sign languages like American Sign Language(ASL) or ISL. Deafness in Bangladesh indicates that the deaf community in Bangladesh urgently needs accessible communication tools or aids. There are millions of individuals with hearing disabilities in Bangladesh who face challenges in communication, education and employment due to the absence of robust support for sign language[1]. Organizations such as the Center for Disability in Development (CDD) have continuously emphasized inclusive solutions, particularly in the area of sign language translation technology.

Automated Sign Language Translation Systems: An early attempt was discussed to translate Bangla into sign language which can adopt from still images and gestures[2]. The system had limitations so that it wasn't capable of processing continuous speech, but it laid the foundation for more sophisticated tools by identifying challenges in converting spoken Bangla to BdSL gestures as a starting point. The study found that a dynamic avatar-based system could enhance flexibility and accuracy.

Global Research in Sign Language Translation: Several research efforts in other regions have successfully established a sign language translation system. In India, Goyal and Goyal [3] devised a synthetic animation system for ISL that replicates human gestures through avatars. Similarly, for Arabic Sign Language, a 3D avatar-based system was developed. Aliwy and Ahmed [4] discussed improved communication through more natural and realistic gestures.

In 2024, A study proposed a multilingual sign language foundation model that eliminates the traditional reliance on gloss annotations. Using a large-scale transformer-based architecture, the model directly maps sign language video sequences to spoken text across multiple sign languages. The gloss-free approach reduced annotation requirements and improved cross-lingual generalization between sign languages such as ASL. Results showed that this model consistently outperformed gloss-based baselines. However, the system is highly resource-intensive and relies on large multilingual datasets, which limits its immediate application to under-documented sign languages like BDSL [27].

In 2024, another paper introduced a hyperbolic contrastive learning framework for sign language translation that leverages the hierarchical and geometric structure of sign gestures. Instead of traditional

Euclidean embeddings, the system uses hyperbolic space to represent skeletal keypoints, improving the ability to capture subtle differences in hand movements. The approach reduced dependency on raw RGB video, which also enhances signer privacy. Experiments showed superior translation performance on benchmark ASL datasets. However, the system has not yet been tested on low-resource sign languages such as Bangla and scalability to complex sentence structures remains a challenge [28].

In 2023, SignDiff introduces a dual-conditioned diffusion model designed to generate skeleton-based American Sign Language (ASL) pose sequences from text input. It includes a Frame Reinforcement Network (FR-Net) for enhanced alignment of text lexical symbols to dense pose frames. Evaluated on How2Sign and PHOENIX14T datasets, SignDiff achieved state-of-the-art BLEU-4 scores and higher SSIM image-quality metrics (~10 percentage points). Limitations include computational intensity and the reliance on skeletal pose output rather than full animations or avatars [29].

In 2022, A research work named as SignGAN ("Everybody Sign Now") is a neural model that generates photo-realistic sign language videos from text. It uses a transformer with a Mixture Density Network to translate text into skeletal pose sequences and then a pose-conditioned human synthesis network to render realistic visuals. The model incorporates a keypoint-space loss to enhance hand quality and supports controllable avatar appearance. It outperformed earlier skeleton-only and avatar-based baselines in both metrics and user perception tests. Limitations include heavy reliance on large datasets and the potential for overfitting to specific signer styles [30].

In 2021, some Researchers at Punjabi University, Patiala developed a system that converts spoken Punjabi into Indian Sign Language (ISL) using speech recognition, NLP and synthetic avatar animation via the HamNoSys and SiGML notation framework. It supports Punjabi, English and Hindi inputs and exists as both a web and mobile application serving communication needs and sign language education. A mobile app is expected by year-end. While highly inclusive in its multi-language support and integration of manual/non-manual features, the system may be limited by the breadth of its ISL sign dictionary and the complexity of regional linguistic variations [31].

In the same year another research presented a real-time Bangla Sign Language translation system that utilizes Mediapipe Holistic for skeletal KeyPoint extraction and an LSTM-based model for classification. The system was capable of mapping known BdSL gestures into corresponding output as text. Real-time isolated signs are found to be 94% accurate. The experiment verified the efficacy of pose-based deep learning for BdSL applications. But the system still had trouble keeping up with continuous signing, complex grammar and natural sentence structures. It was tested under laboratory conditions. This raises questions about its robustness in a real-world deployment [32].

In 2020, Rahaman et al.[33] a voice and text translation system. It was developed to transcribe the Bangla text to BDSL signs. The system processed linguistic patterns. This system is also responsible for converting these patterns into sign gesture outputs according to its sign language database. It had a great success for the spoken voice input with 96.03%. For this system, text translation had 100% accuracy. Their system has a rule-based architecture. That's why it was less flexible when it worked with complex sentence structures. Its scalability became sort of restricted in different linguistic settings.

Significant work has been done via the Speech Input and Grammar Markup Language (SiGML) and the HamNoSys. The contributions of this thesis are to help develop a complete sign language synthesis system. The SiGML notation is shown by research[12]. When it was integrated into the HamNoSys, it gave several options for expressing sign language gestures. This work was later developed on in the VISCast Project [8]. The system was therefore designed for capture and animation. This was attempted through animated 3D avatars. These systems offer a strong structure. They can be used for creating Bangla Sign Language (BdSL).

The existing literature remains empty even with all of the recent advances. Based on recent research, currently there are still a handful of real-time translation systems for continuous Bangla speech and a few complete 3D avatar-based BdSL solutions. Most available BdSL recognition systems concentrate

on isolated signs such as words, letters, or numbers, but fall short in delivering complete, real-time sentence translation. Our research proposes a real-time Bangla voice-to-sign translation system that employs HamNoSys and SiGML for generating 3D avatar animations to address those shortcomings. This system is designed to overcome the challenges of earlier approaches by managing continuous speech and dynamic gestures while enhancing accuracy, synchronization and user adaptability. We have created more words and letters than the previous works and a web base solution which can convert Bangla speech into sign language in real-time. Using the words we can create a large number of sentences. Other works was not able to convert a voice into sign language animation in real-time. We have cover this gap by our real time 3D avatar generation system.

Table 2.1: Summary of voice to sign conversion previous works.

Title	Language / Focus	Methodology	Key Findings	Limitations
Tessa: Speech-to-Sign Avatar for Postal Services[1]	British Sign Language (BDSL)	Rule-based speech-to-sign mapping with virtual avatar	First system to deploy real-time speech → sign animation in UK postal services	Limited vocabulary; unrealistic avatar animations
SLAIT – Sign Language Animation in Text-to-Sign Systems[34]	ASL	Natural Language Processing (NLP) + HamNoSys → Avatar	Mapped text → HamNoSys → avatar animation	Linguistic accuracy varied; limited contextual handling
ASL Translator App with Avatars[35]	American Sign Language	Speech recognition + dictionary-based avatar animation	Provided real-time mobile text/speech → ASL avatar	Small lexicon; lacked grammar handling
Rule-Based Bangla Voice/Text to BdSL Sign Generator [33]	Bangla Sign Language	Rule-based NLP + BdSL database + avatar animation	Achieved 96% voice → sign accuracy; 100% for text input	Poor handling of complex sentences; not scalable
Arabic Sign Language 3D Avatar System [36]	Arabic SL	NLP + 3D avatar animation	Produced realistic animated signing from text input	Dataset limitations; focused on words/phrases
Deep Learning-Based Text-to-	Indian SL (ISL)	Transformer-based NMT +	Improved natural signing with context	Heavy computation; still research stage

Sign Generation[37]	avatar synthesis	preservation

1.3 SIGN TO VOICE AND TEXT CONVERSION

Initial systems for sign language recognition mainly designed for recognizing static gestures like letters or numbers. They used traditional image processing and algorithms for example SVMs(Support Vector Machines) and KNNs(k-Nearest Neighbors). Deep learning helped to bring major improvements with CNN-based(Convolutional Neural Networks) approaches in this field. Hybrid models combining DNNs (Deep Neural Networks) with HMMs (Hidden Markov Models) have also advanced recognition and performed well in controlled environments. They also achieved higher accuracy. Compared to other sign languages, technological solutions remain limited.

The Hybrid Efficient Convolution (HEC) model is presented by Chowdhury et al.[14]. It was basically designed for Isolated Dynamic Sign Language Recognition. They used EfficientNet-B3 and a customized dense layer. This provided better performance in a variety of environments. The authors also created a new data set from scratch. The name of the data set is BdSL OPA 23 GESTURES. It consists of 6000 videos of 100 Bangla signs. The dataset is favorable for training. The HEC model had a 93.17% accuracy rate. It outperformed well-established and older architectures such as AlexNet and ResNet-50. Some key contributions are the introduction of a hybrid CNN framework, data set variety, and improvements in feature extraction methods. Without these strengths, the model struggles with more complex sign distinctions, limitations in substantial background variety, and is limited to isolated sign recognition only. The future of this work is spent optimizing the computational efficiency further. It will include a greater variety of dataset variations.

In the last few years, great progress has been made in creating resources for BdSL. Islam et al. [38] initiated Ishara-Lipi. This is a multipurpose dataset of isolated BDSL characters and words. This dataset acts as a crucial resource for constructing BDSL recognition and translation systems. More recently, another dataset has expanded its scope. Name of the dataset is BDSL 49[39]. It provided an extensive collection of Bangla signs and enabled more accurate recognition systems for continuous BDSL detection but still not enough to explore a language. The recognition accuracy for BDSL with the help of deep learning applications have further improved accuracy[40]. They proposed a model combining MobileNetV2 and a conditional DCGAN. This Model is to build a convenient, real-time BDSL recognition system. Similarly, the study by Akash et al.[41] induced an action-recognition-based model which can translate BDSL gestures into complete sentences in real-time, exhibiting significant progress in the system's usability.

In 2024, Rahaman et al.[42] build a real-time system. The system is computer vision-based. It recognized Bangla sign language gestures. Three key linguistic features are used in their model. They are: hand shape, movement position extracted through techniques such as the Binary Window-Grid Vector (BWGV), Normalized Outer Boundary Vector (NOBV) and Adaptive Kalman Filter (AKF). They trained on a dataset of 100 Bangla gestures, the system achieved an accuracy of 95.43% with a processing speed of 56.013 milliseconds per frame. While their research increased movement tracking and hand-shape recognition efficiency, it fell short in incorporating all linguistic features, leading to potential misclassification of more complex gestures. Future work aims to extend linguistic feature integration and further improve real-time accuracy.

Rahaman et al.[20] induced ConvNeural in the same year, a 6 layer convolutional neural network (CNN) model aimed at recognizing Bangla sign language numbers and alphabets. The model architecture included 4 convolutional layers dedicated to feature extraction, followed by two fully connected layers for classification. To train ConvNeural, the researchers build two datasets BdSL OPSA22 STATIC1 and BdSL OPSA22 STATIC2 and the dataset comprising over 30,000 images with different backgrounds.

This model demonstrated superior performance, achieving accuracy rates of 98.38% and 92.78%, surpassing several pre-trained models. Neglect its high accuracy and effective integration of deep learning techniques, ConvNeural encountered difficulties in differentiate between similar signs, partly due to a small gesture diversity in the dataset. While the large dataset supported robust model training, the lack of variability in gesture representation posed challenges. Moving forward, improvements will focus on increasing real-time recognition capabilities and expanding dataset coverage to reduce misclassification errors.

In 2023, Islam et al.[43]introduced MVBDSL-W50, a Bangla gesture dataset comprising 4,000 videos representing 50 words, performed by participants aged twenty to thirty. The recordings were conducted against a green screen to maintain visual consistency; however, this controlled environment limited the dataset's applicability in real-world scenarios due to the absence of background variation. Previous research in Bengali sign language recognition mainly dependent on computer vision-based techniques, highlighting the need for more different background and context-rich datasets like MVBDSL-W50 to bridge the gap between controlled settings and practical applications.In 2023, Abedin et al. [44] provided a system for Bangla sign language recognition from video sequences using a concatenated BDSL network. Their approach connected multiple networks to increase feature extraction and was trained on a dataset of 1,000 videos encompassing 100 different signs. Their model achieved 98.7% accuracy. It had some limitations, such as not accounting for different complex backgrounds. This could affect real-world performance. The dataset used was relatively small compared to other studies. It can limit broader generalization and robustness.

In 2023, Das et al.[40] and his colleagues introduced a hybrid model. The model combined Deep Convolutional Neural Networks (DCNNs) with Hidden Markov Models (HMMs). It used for sign language recognition. The DCNN technique was responsible for filtering visual features and the HMMs were used to model the temporal sequences of gestures. The system was tested on the RWTH-PHOENIX-Weather 2014T and ASLLVD datasets. For the testing, it achieved Word Error Rates (WER) of 0.172 and 0.079 respectively. This system had performed strongly, yet the model could not challenge related to variations of features. The features were lighting, backgrounds, age, gender, etc. These limitations could affect its robustness in real-world environments.

A method was found by Kothadiya et al. [45] The technique of their research was made for recognizing sign gestures from video sequences. They combined Deep Neural Networks (DNNs) with Hidden Markov Models (HMMs).The DNN was employed to extract features from individual frames. The HMM used for capturing the sequence of gestures, via the Viterbi algorithm for decoding. The system showed very high performance. It achieved 97% accuracy on the ASLLVD dataset and 98% on the Indian Sign Language Dataset (ISLD). Despite these very high accuracy rates, the method did not account for variations in background, skin tones, or lighting conditions, which limited its robustness and applicability in diverse real-world environments.

In 2021, Ahammad et al. [46] invented a model. The research model was for identifying signs using Support Vector Machines (SVMs) and k-Nearest Neighbors (KNNs).The system used feature extraction followed by shape analysis using Hu and Zernike moments. It achieved an accuracy of 98% with SVM and 97% with KNN on a dataset comprising 100 videos. Although the system provide high accuracy, it could not handle different background conditions, limiting its generalizability and effectiveness in real-world applications.

In 2020, Kamrul and Sharmin [47] build an Android app. The app supposed to provide two-way communication between hearing-impaired and Bangla-speaking persons. Their app used Google's speech recognition. It can convert spoken Bangla into animated sign language and allows hearing-impaired people to type Bangla text via a custom keyboard. It was tested in real-life scenarios. The system achieved 88% accuracy in voice-to-sign conversion and 91.67% in sign recognition. Similar to other models, it also faced challenges, including varied pronunciation and difficulties with 3D motion

signs. Future improvements aim to add more vocabulary and enhance the keyboard interface.

In 2017, Rahaman et al.[48] proposed a real-time system using Haar cascade classifiers and skin-color segmentation for hand gesture recognition. It was for recognizing Bengali alphabet and numeric signs. Features were extracted as row vectors from binary images. They were classified using a K-Nearest Neighbors (KNN) algorithm. The system was trained on 3,600 images of 36 letters and 1,000 different images of 10 numbers. It achieved about 92.04% accuracy for the alphabet's images and 96.67% for the number's images. But it is limited in varied lighting and backgrounds. The system struggled with similar hand postures and skin-colored background interference. So that it indicates areas for future improvement.

Despite recent progress in BdSL recognition, there are still some limitations that exist. Most of them are focused on isolated signs, which contain individual words, letters or numbers. Models such as the Hybrid Efficient Convolution (HEC), ConvNeural, and concatenated BD-SL network have shown high accuracy rates. They also have problems with complex differences in gestures and dynamic backgrounds with different lighting conditions. Many datasets are small. The datasets can only cover a small number of words, which may be recorded in a static manner. It can withstand their capability to represent the full complexity of the language and its use in real-world scenarios. Our research is able to eliminate these gaps by creating a BdSL recognition system using real-time. The system will be in a position to manage continuous gestures and dynamic environments. By integrating advanced feature extraction techniques using Mediapipe. We have extracted keypoints from a signer and then used the keypoint for training. This technique reduces the computational time. We have also contributed to dataset diversity to include a broader vocabulary and varied environmental conditions. Our system will improve recognition time, and this will help to recognize signs in real-time. This approach can reduce the limitations of isolated gesture recognition and is comprehensive and practical.

Table 2.2: Summary of sign to text and voice conversion previous works.

Title	Focus/ Dataset	Methodology	Key Findings	Limitations
Hybrid Efficient Convolution (HEC) Model for Dynamic Sign Language Recognition [49]	BdSL OPA 23 Gestures dataset (6,000 videos, 100 signs)	EfficientNet-B3 + dense layer hybrid CNN	93.17% accuracy; outperformed AlexNet, ResNet-50	Limited to isolated signs; struggled with complex gestures & varied backgrounds
Real-Time Computer Vision-Based BdSL Recognition [42]	100 BdSL gestures	BWGV + NOBV + AKF for hand shape, movement, position	95.43% accuracy; 56 ms per frame	Missed some linguistic features; risk of misclassifying complex signs
ConvNeural: CNN-Based BdSL Recognition Model [50]	BdSL OPSA22 STATIC1	6-layer CNN (4 conv + 2 FC layers)	98.38% and 92.78% accuracy	Difficulty with similar signs; limited gesture diversity
Concatenated BdSL Network [44]	1,000 videos, 100	Multi-network concatenation for	98.7% accuracy	Small dataset; lacks complex

	signs	feature extraction		backgrounds
Hybrid DCNN-HMM Model for Sign Recognition	RWTH-PHOENIX-Weather 2014T, ASLLVD	DCNN for features + HMM for temporal modeling	WER: 0.172 (RWTH), 0.079 (ASLLVD)	Sensitive to lighting, age, gender, skin tone variations
DNN-HMM for Video-Based Sign Recognition [51]	ASLLVD, Indian SL Dataset	DNN for features + HMM (Viterbi decoding)	97% (ASLLVD), 98% (ISLD)	Lacked robustness to backgrounds/lightin g
Shape-Based Recognition using SVM & KNN [52]	100 videos	Hu & Zernike moments + SVM/KNN classifiers	98% (SVM), 97% (KNN)	Poor generalization under varied conditions
Real-Time Bengali Alphabet & Numeric Recognition [48]	3,600 alphabet images + 1,000 numerals	Haar cascade + KNN	92.04% (alphabets), 96.67% (numbers)	Lighting & background interference; similar signs problematic

1.4 RESEARCH GAP

Even though other global sign languages such as American Sign Language (ASL), Indian Sign Language (ISL) etc, are improving inspiringly. By doing some research about automated sign language systems, it is obvious that this field is progressing considerably. But regretfully, Bangla Sign Language (BdSL) is still not developed enough compared to other widely studied languages (ASL or ISL). An analysis of various reports on deafness in Bangladesh shows that there are millions of individuals with hearing impairments, but BDSL-related technological progress is still not up to date to overcome the communication limitations of deaf people. The lack of these has been described above, and it is an emergency for the Bangladeshi deaf community to have useful communication aids. Existing BdSL recognition systems can not handle continuous gesture recognition in real-time. One of the main limitations is that it deals only with isolated signs. Some techniques, such as CNNs, HMMs, and hybrid models, can achieve high accuracy rates in controlled environments. But they still do badly in complex real-life situations when it comes to generalization. These models are found not to be very robust due to their failure to handle moving background, illumination, and complex gesture patterns. BdSL recognition was performed against a database with small diversity and small size. Some other resources (Ex, Ishara-Lipi and BdSL 49) were consulted and used for support. They are not sufficient to represent the full potential of the BdSL language. Database collections are in controlled conditions with no background or natural light variation levels. The reason is that it limits the ability of models to be used in other or complex environments. Previously used rule-based detection systems were shown to have a high accuracy. Like other models or systems that had complex syntactic rules, they had a limited vocabulary. This bar prevented them from reaching their potential growth. Other sign languages have even improved sign language synthesis. The combination of 3D avatar-based systems with standard models such as

HamNoSys and SiGML, therefore, made possible the implementation of a more flexible rendering of signs. However, the concept of avatar-based solutions that can process continuous sentences and dynamic gestures in real-time is still lacking in BdSL. Research gaps are associated with the necessity of developing advanced BdSL recognition systems. Some systems are unable to carry out the real-time recognition and sign translation of continuous speech and support any type of unknown environments and complicated sign language motions.
New feature extraction and 3D avatar technologies have been introduced to gesture recognition, though the amount of resources for connecting the isolated components is limited, and overloading of data sets for the common task is required. A survey of the Bangla Sign Language (BdSL) translation system literature has been reported. The review is an examination of previous studies that are related to our current topic. It describes the methodologies and processes involved for both voice-to-sign and sign-to-voice translation technologies. The review draws the connection between previous research and this research. This chapter contains the development of sign language technology through advances in both 3D avatar animation and deep learning models for gesture detection. The existing Bangla Sign Language translation systems are examined in this research. Special attention is given to real-time communication within Bangladesh. The literature review builds a strong foundation for the ongoing research by identifying the key challenges. Different approaches used in prior works are critically assessed to highlight their level of effectiveness. It points out potential applications for developing improved, real-time solutions for the Bangla-speaking deaf community.

1.5 SUMMARY

Current Bangla Sign Language (BDSL) recognition have focused on limited signs recognition using deep learning techniques like CNNs, HMMs and hybrid models, achieving high accuracy in controlled environments. However, the current systems struggle with complex gestures, dynamic backgrounds and continuous real-time transformation. Existing datasets, such as Ishara-Lipi and BDSL 49, are small in size and diversity, which is not enough to cover a whole language. Unlike other sign languages, BDSL lacks robust 3D avatar-based solutions and comprehensive tools for continuous gesture recognition. Addressing these limitations requires increasing dataset diversity, improving model robustness and integrating advanced technologies for dynamic, real-time BDSL transformation. A summary table for our provided reference is provided below:

Table 2.1: Summary table for both voice to sign and sign to voice conversion.

Title	**Dataset**	**Method**	**Findings**	**Limitations**
HamNoSys to SiGML Conversion for Sign Language Automation [9]	HamNoSys and SiGML framework	Conversion of HamNoSys notation into SiGML scripting for automated gesture generation	Enabled standardization of sign language notation for 3D avatar animation	Lacked language-specific adaptability for Bangla Sign Language (BDSL)

ViSiCAST Project: 3D Avatar-Based Sign Language Generation [10]	ViSiCAST system dataset	Developed tools for capturing, animating and transmitting sign language gestures using 3D avatars	Created a robust framework for real-time sign language translation using avatars	Focused on European sign languages, not Bangla Sign Language
Hybrid Efficient Convolution (HEC) for Isolated Dynamic Sign Language Recognition [16]	BdSL OPA 23 GESTURES (6,000 videos, 100 Bangla signs)	EfficientNet-B3 with a custom dense layer	Achieved 93.17% accuracy, surpassed AlexNet and ResNet-50	Struggled with complex gesture distinctions, lacked background variety, limited to isolated gestures

Title	**Dataset**	**Method**	**Findings**	**Limitations**
Real-time Bangla sign language recognition [17]	100 Bangla gestures	Computer vision-based approach using NOBV, BWGV and Adaptive Kalman Filter	95.43% accuracy, processing speed of 56.013 ms per frame	Did not incorporate all linguistic features, misclassification of complex gestures
ConvNeural: Six-layer CNN for BDSL recognition [18]	BdSL OPSA22 STATIC1 & STATIC2 (30,000+ images)	Four convolutional layers for feature extraction, two fully connected layers for classification	98.38% and 92.78% accuracy, outperformed pre-trained models	Struggled with similar gestures, dataset lacked diversity
MVBDSL-W50 Dataset for Continuous BDSL Recognition [19]	4,000 videos for 50 Bangla words	Video dataset collection for gesture recognition	Addressed need for context-rich datasets in BDSL recognition	Controlled environment (green screen) limited real-world applicability
Bangla Sign Language Recognition using Concatenated BD-SL Network [20]	1,000 videos encompassing 100 different signs	Deep learning-based concatenated BD-SL network	Achieved 98.7% accuracy, improved feature extraction	Did not consider cluttered or complex backgrounds, dataset size limited.
Hybrid DCNN + HMM Model for BDSL Recognition [21]	RWTH-PHOENIX-Weather 2014T and ASLLVD	Deep Convolutional Neural Networks (DCNNs) with	Achieved low Word Error Rates (WER) of 0.172 and 0.079	Did not address lighting variations, background complexity, or skin

	datasets	HMM		tone differences
DNN + HMM for Isolated BDSL Gesture Recognition [22]	ASLLVD and Indian Sign Language Dataset	Deep Neural Networks (DNNs) and HMMs	Achieved 97% accuracy on ASLLVD and 98% on ISLD	Did not account for background, skin tone and lighting variations

Title	**Dataset**	**Method**	**Findings**	**Limitations**
Android-Based Bangla Sign Language Communication App [24]	Live testing in real-life scenarios	Google Speech Recognition for voice-to-sign and text-based input for sign-to-text	88% accuracy in voice-to-sign, 91.67% in sign recognition	Challenges with pronunciation variations, 3D motion signs and limited vocabulary
Real-time Bengali Alphabet and Numeral Sign Recognition [25]	3,600 images of 36 alphabets and 1,000 images of 10 numerals	Haar cascade classifiers and K-Nearest Neighbors (KNN)	92.04% accuracy for alphabets, 96.67% for numerals	Struggled with similar hand postures and skin-colored background interference

CHAPTER 3

METHODOLOGY

3.1 INTRODUCTION

In this chapter, we will discuss the methodology for building a real-time Bangla Sign Language (BSL) recognition and 3D avatar generation system. This process contains two main tasks: the first one is translating Bangla voice into 3D animated avatars, and the second one is recognizing Bangla sign language gestures. The tasks will be explained in two separate sections. The first section focuses on translating Bangla voice into 3D avatar-based sign language animations. Using frameworks like HamNoSys and SiGML, the system will convert spoken Bangla into dynamic, real-time sign language animations, enhancing communication accessibility. The second section focuses on sign language recognition. The methodology includes data collection, preprocessing, feature extraction, model development, and performance evaluation.

3.2 METHODOLOGY OVERVIEW

The given figure illustrates a bidirectional communication system that translates Bangla speech to Bangla Sign Language (BDSL) and vice versa. The process is divided into two parts: Bangla speech input to sign language output, and sign language gestures to Bangla text/voice output. In the speech-to-sign translation, Bangla speech is captured via a microphone or audio file and converted into text using the Google Speech API. The text undergoes tokenization, parsing, and lemmatization before being analyzed to determine if the corresponding sign exists in the Bangla Sign Language database. If the word is found, it is mapped with our database and retrieved from the Bangla SigML Database (containing 3,000 words). The corresponding sign is then animated using a 3D avatar for output. However, if the word is not found in the database, the system decomposes it into individual letters and generates a letter-based sign animation, ensuring that even unknown words can be communicated. Simultaneously, the second part mainly contains with sign language gestures dataset collected from video data. The model will extract frames from video (30 frames per second), gestures, and preprocesses them for deep learning training. The dataset is split into training and testing sets, where 80% data will be used for training and 20% data will be used for testing. The different deep learning models, such as Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs), will be used for feature extraction and gesture recognition. Our trained models will then be compiled and validated through model evaluation, ensuring accurate sign language recognition. After the model processes an input gesture, it converts it into Bangla text or converts it into speech, providing a real-time communication connection between hearing and hearing-impaired individuals. This comprehensive methodology connects linguistic processing, deep learning, and 3D animation, ensuring a scalable and efficient solution for improving accessibility for the deaf community.

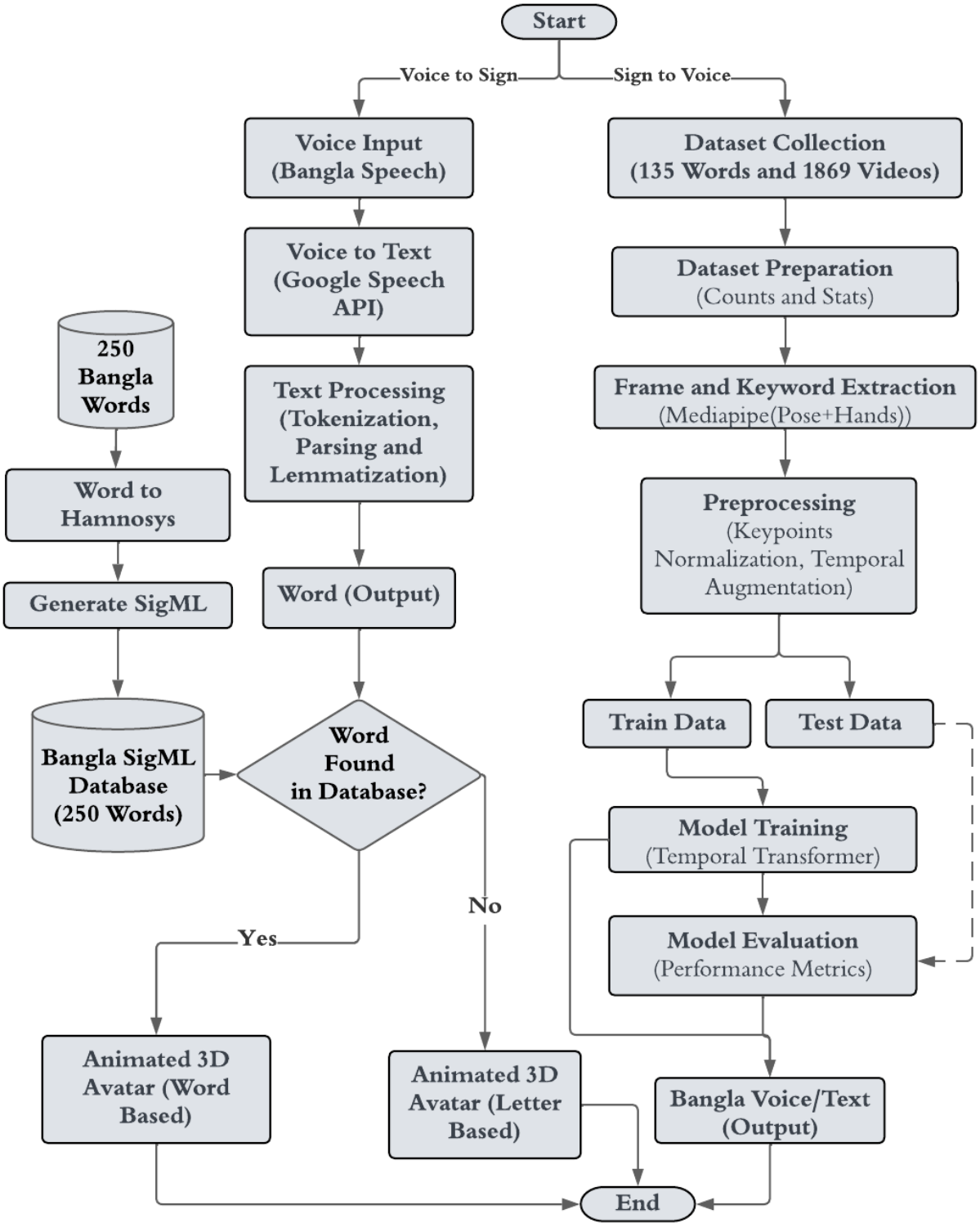


Figure 3.1: Workflow Diagram of Our Research Work.

We have divided our whole work into two different parts: one is voice-to-sign language conversion, and the one is sign language-to-voice conversion, and they are discussed in detail in Sections 3.3 and 3.4, respectively.

3.3 VOICE TO SIGN LANGUAGE CONVERSION

Our proposed system can provide bidirectional communication. In one direction, a hearing individual will speak in Bangla using a microphone, which is then translated into Bangla text, and the text will be converted into Bangla Sign Language (BSL) and animated by a 3D avatar, as illustrated in Figure 3.2.

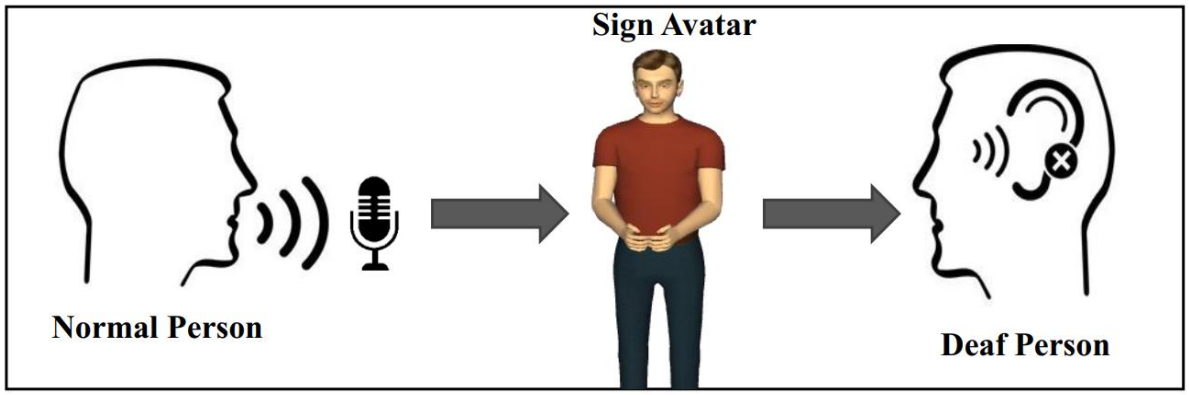


Figure 3.2: Voice-to-Sign Language Translation.

In figure 3.3. we can see our proposed system starts with voice input, where a user can speak Bangla speech, which will be captured by a microphone and processed using the Google Cloud Speech API, converting it into text. The text is then sent through tokenization and parsing, and it will break it down into words, followed by lemmatization to reduce words to their base forms. After completing the text preprocessing, the text will target to Bangla Sign Language using the HamNoSys notation. The system generates SiGML files that encode the sign language gestures.

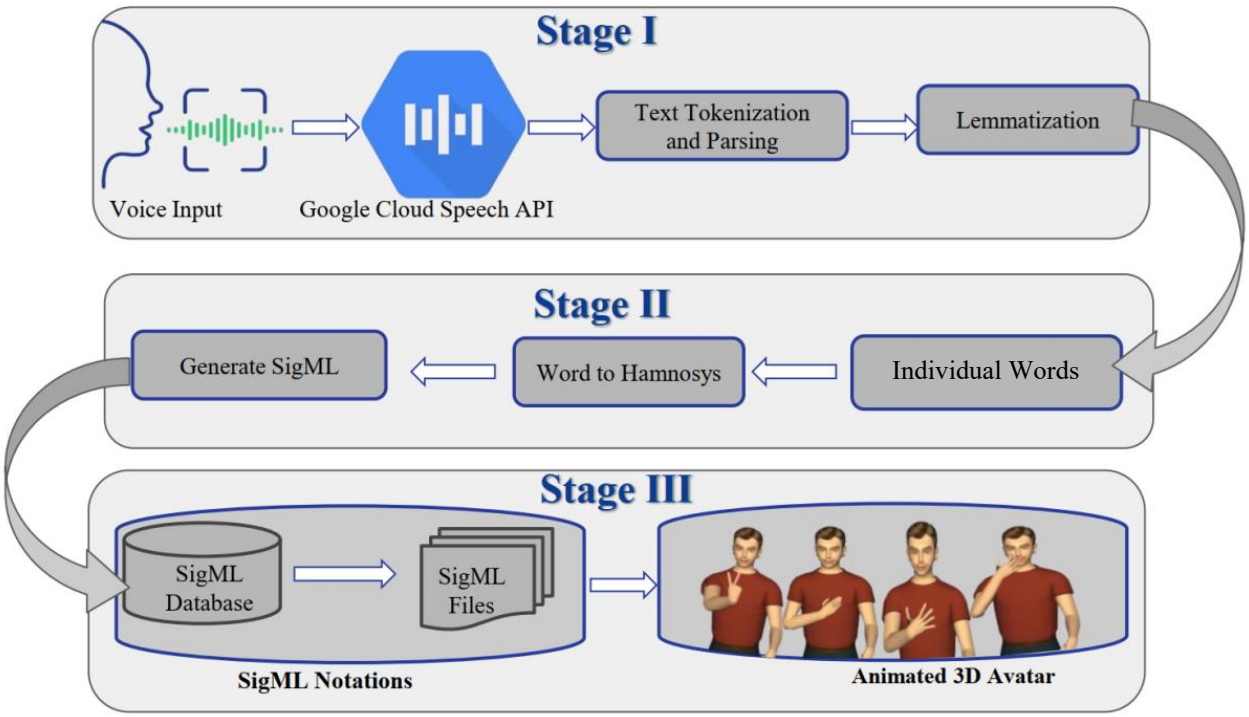


Figure 3.3: Workflow diagram for voice-to-sign language avatar generation.

Then, finally, these SiGML files are used to animate a 3D avatar by using a web-based platform, which visually performs the Bangla Sign Language gestures in real-time. Our proposed methodology ensures an efficient and accurate voice to sign language translation technique.

3.3.1 Stage I: Speech to Text Conversion

The figure 3.4 shows that in Stage I, the system captures audio input in various formats and real-time speech. This ensures high-quality data through noise reduction. The Google Cloud Speech API transcribes spoken Bangla into text, which is then tokenized, parsed, and lemmatized. These techniques helps to facilitate accurate translation into Bangla Sign Language gestures, preserving grammatical structure and contextual meaning.

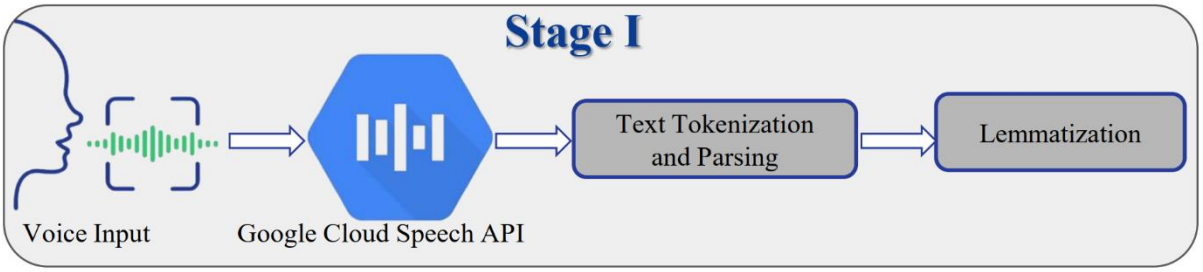


Figure 3.4: Stage I (Speech Processing and Text Conversion)

Voice Input: In this step, the system will accept audio input in multiple formats, such as WAV, MP3, and FLAC. These are the commonly used file types. The system will capture real-time voice input directly from a microphone, allowing for dynamic user interaction. The voice data collected at this stage will serve as the primary input for the system. It will then undergo processing for further translation into Bangla Sign Language. Noise reduction techniques will be applied to minimize background interference and ensure high-quality input. These are the foundation for accurate speech to text conversion and subsequent stages of sign language generation.

Google Cloud Speech API: The Google Cloud Speech API plays a vital role in converting spoken Bangla into text. This API uses advanced machine learning models specifically designed to support Bangla. These ensure accurate speech-to-text conversion. After the speech is captured, the API processes it and transcribes the spoken words into written text. It is optimized to handle the unique accents and dialects of Bangla. They are providing precise transcription even with variations in pronunciation. This transcribed text then forms the foundation for the next steps in the system. Some techniques, such as text parsing, lemmatization, and gesture mapping, are used here[53].

Text Tokenization and Parsing: After the transcribed Bangla text is systematically decomposed into smaller. It is a manageable component to facilitate detailed analysis. Tokenization serves as the initial step. In this step, the text is divided into individual words or meaningful units known as tokens. The table below shows the text tokenization. This structural analysis provides insights into the relationships between the tokens. This ensures the overall meaning of each sentence is preserved.

Table 3.1: Sentences that have been converted into tokenized parts.

Sentence	Tokenization		
আমি স্কুলে যাচ্ছি	আমি	স্কুলে	যাচ্ছি
দুইটি সবুজ পেয়ারা	দুইটি	সবুজ	পেয়ারা

Lemmatization: The Bangla text that is being transcribed is reduced systematically to its base or dictionary form. This is known as the lemma. It helps to facilitate an accurate understanding of the meaning of words in the context of a sentence. Lemmatization is one of the important steps that comes after tokenization. Each token is converted into its raw form. This process helps to increase the ability to analyze the text by grouping the different inflected forms of a word. This makes it possible to interpret the underlying meaning much clearly. The following table summarizes the tokens and their corresponding lemmas for each sentence:

Table 3.2: Tokens and their corresponding lemmas for each sentence.

Sentence	Token	Lemma
আমি স্কুলে যাচ্ছি	আমি	আমি
	স্কুলে	স্কুল
	যাচ্ছি	যাওয়া
ছেলেরা খেলার জন্য প্রস্তুত।	ছেলেরা	ছেলে
	খেলার	খেলা
	জন্য	জন্য
	প্রস্তুত	প্রস্তুত

This structural analysis gives some information about the relationships between the tokens. It is to make sure that the overall meaning of any sentence is not lost. By implementing tokenization and lemmatization, both our systems are able to translate even a complex Bangla sentence to its corresponding Bangla Sign Language gestures. This ensures the grammatical integrity and contextual meaning are preserved throughout the translation process. This maintains grammatical integrity also the contextual meaning throughout the translation process.

3.3.2 Stage II: Word to SigML Generation

The Data Preprocessing stage cleans and structures Bangladeshi Bangla text for accurate sign language translation. This is done by removing unnecessary words and correcting errors. In the Word to HamNoSys Conversion stage, each word is mapped to hand movements using HamNoSys. Then it encodes hand shapes and movements. Finally, the Generate SiGML file step converts HamNoSys notations into SiGML scripts. This enables 3D avatars to animate the corresponding sign language gestures. Figure 3.5 shows the process of word to SigML file generation.

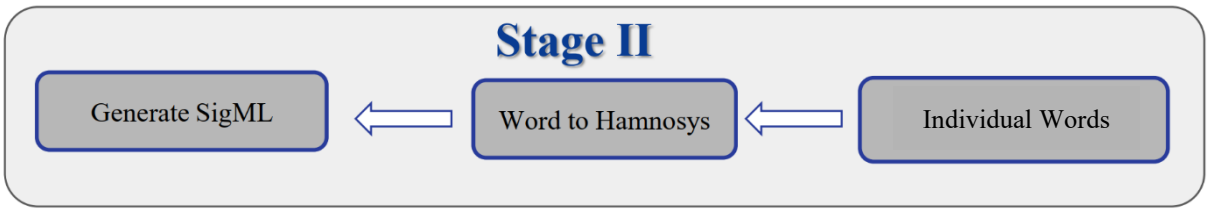


Figure 3.5: Stage II (Word to SigML Generation)

(i) Individual Words

In this step, unnecessary or irrelevant words, such as fillers or noise. These are removed to enhance the clarity of the text. Any punctuation, grammatical errors, or inconsistencies in the text are addressed to improve the accuracy of the sign language generation. Preprocessing also includes the standardization of the text format. This ensures that the input is consistent and properly formatted for the next stages. This step is essential to provide individual raw words extracted from the sentence.

(ii) Word to Hamnosys

HamNoSys is a standardized system used to describe the physical components of sign language, including hand shapes, movements, orientations, and positions. It serves as a bridge between written language and visual gestures by encoding how a word is represented in sign language through symbols. To get Bangla HamNoSys notation, a predefined database or dictionary of Bangla Sign Language (BSL) gestures is referenced. Each word in Bangla is associated with a specific hand movement, facial expression, or body gesture that matches its meaning in BSL. There are a large number of hand position notations in HamNoSys; their summary is given below [54].

Symmetric: Symmetry Operator (Optional) is typically represented with two dots to signify both hands are used to do the sign or specify if the non-dominant hand is utilized.

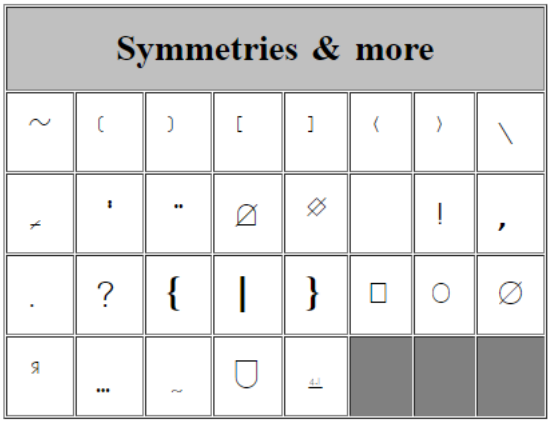


Figure 3.6: Symmetry Operator in HamNoSys Notation.

Handshape: Hand form refers to the characteristics of the hand, such as the number of fingers that should be extended, the degree to which they should be bent, and the placement of the thumb. The most popular symbols in this category are an oval with lines to represent extended fingers, curved lines to indicate the proper finger bend (with numbers 1 through 5 designating

which finger to bend), and any marks or fractures inside the oval to represent the distance between the fingers and palm.

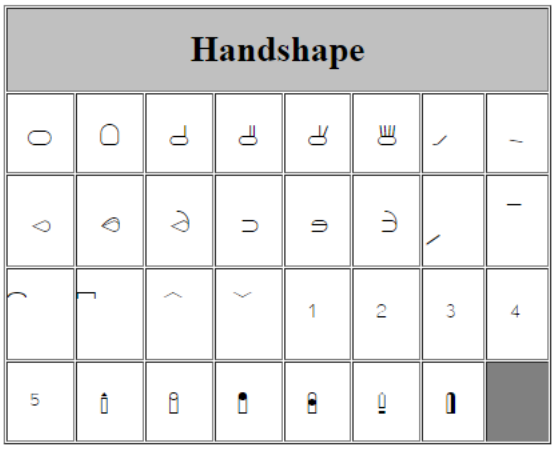


Figure 3.7: Handshape in HamNoSys notation.

Hand Position: The hand position is also part of the same segment as (3). These, for instance, display the direction in which the extended fingers are pointing or the position of the palm. In the context of palm orientation, the symbols consist of little carrots indicating certain or more advanced angles and pointing to something else, along with narrow ovals with one thick side facing distinct directions.

Hand Location: When signing, hand placement refers to the relative position of the hand about the body. Certain symptoms appear near the chin or temple, while others are in front of the sternum or shoulders. The locations of are shown by a black square positioned in the direction of the characteristic, using symbols that often resemble the bodily parts being described.

Movement: It indicates any movement in the signals, such as a sign moving from the left side

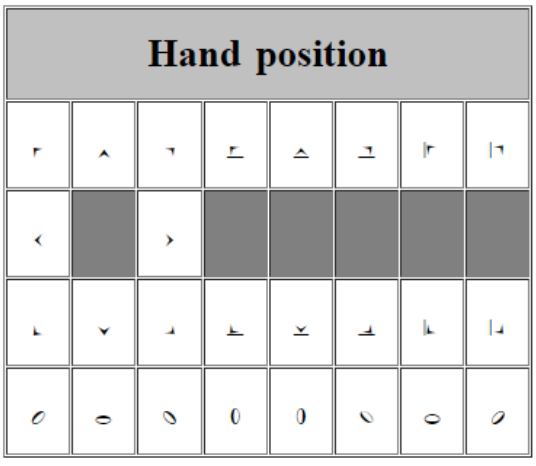


Figure 3.8: Hand Position in HamNoSys notation.

Figure 3.9: Hand Location in HamNoSys notation.

of the sternum to the right or a finger pointing up and then down. These are often intricate,

using symbols that resemble the arrows and carrots in hand orientation, together with lines that might represent circular, wavy, or straight motions. Additionally, there are characters like parentheses or brackets that divide motions that are sequential or cohesive.

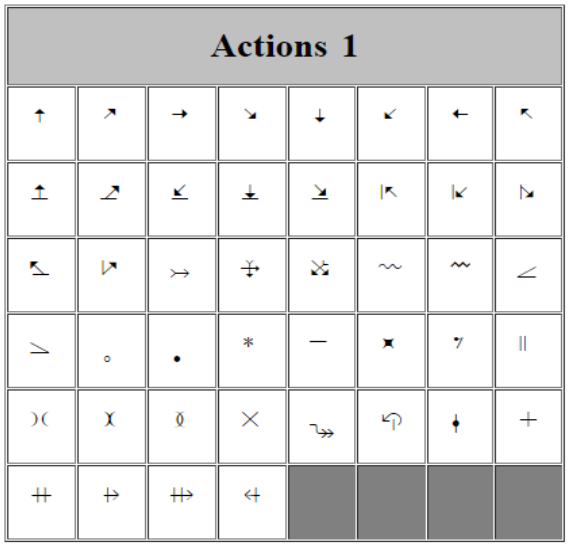


Figure 3.10: Straight Movement.

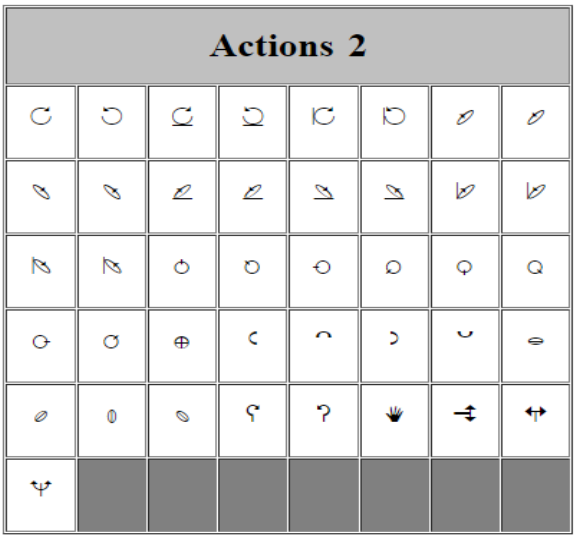


Figure 3.11: Circular Movement.

(iii) Generate SigML File

SiS-Builder is a web-based tool developed under the DICTA-SIGN project to convert HamNoSys notation into SiGML scripts for animating virtual avatars. Users can input HamNoSys characters of a sign, and the tool automatically generates the corresponding SiGML transcription, enabling sign language gestures to be performed by a 3D avatar. The tool translates the hand shapes, movements, and orientations from HamNoSys into XML-based SiGML, ensuring accurate representation of sign language gestures.

For example, the word “mug” in Bangla Sign Language can be converted into SiGML by first generating its HamNoSys notation, which is then processed by SiS-Builder. The resulting SiGML script controls the avatar’s gestures, animating the sign for "mug."

Table 3.3: SiGML XML file for the word (পান করা).

```
<sigml>
<hns_sign gloss=" পান করা">
        <hamnosys_nonmanual>
        </hamnosys_nonmanual>
        <hamnosys_manual>
                <hamfist/>
                <hamthumboutmod/>
                <hamextfingeru/>
                <hampalml/>
```

```
                    <hamchin/>
                    <hamseqbegin/>
                    <hamclose/>
                    <hamthumb/>
                    <hamseqend/>
                    <hamreplace/>
                    <hamfist/>
                    <hamthumboutmod/>
                    <hamextfingeru/>
                    <hambetween/>
                    <hamextfingeri/>
                    <hampalml/>
                    <hamchin/>
                    <hamseqbegin/>
                    <hamclose/>
                    <hamthumb/>
                    <hamseqend/>
                </hamnosys_manual>
            </hns_sign>
            </sigml>
```

3.3.3 Stage III: SigML to 3D Avatar

Stage III involves converting HamNoSys representations into SiGML files that guide a 3D avatar in real-time sign language animations. The avatar translates spoken Bangla into Bangla Sign Language gestures, enhancing communication for the hearing-impaired. This web-based application aims to improve accessibility and promote the use of Bangla Sign Language in Bangladesh.

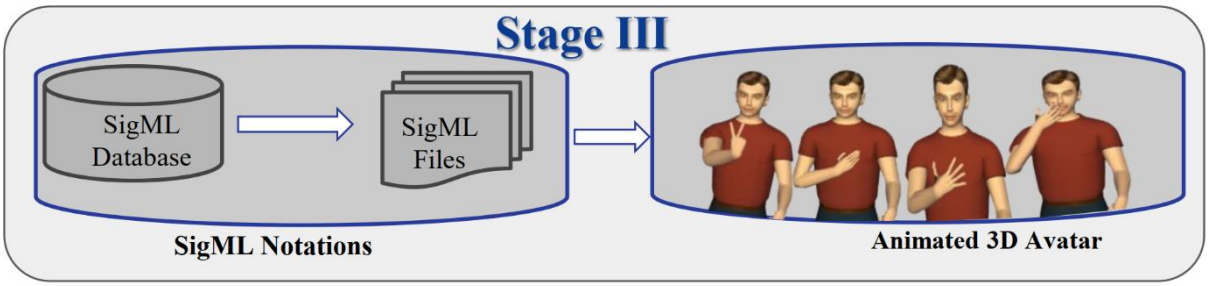


Figure 3.12: Stage III (SigML to 3D Avatar)

(i) SigML Notations

The SiGML Notations step involves converting HamNoSys representations into SiGML files, which are stored in a SiGML database. This database contains pre-defined SiGML scripts for various Bangla Sign Language gestures. When a specific word, such as “mug”, is input into the system, the corresponding SiGML file is fetched from the database, containing all the necessary information, including hand shapes, movements, orientations, and non-manual features like

facial expressions. If a SiGML file for the word does not exist, it can be generated based on its HamNoSys notation. These fetched or newly created SiGML files are then used to control the avatar's animations, ensuring accurate and fluid representation of the sign language gesture.

(ii) Animated 3D Avatar

The Animated 3D Avatar stage is the final, visually impactful part of the system where SiGML files animate a 3D avatar to perform Bangla Sign Language (BSL) in real-time, translating Bangla voice input into sign language gestures for effective communication with the hearing-impaired.

3D Avatar Generation: The avatar is generated using web-based platforms like JASigning or similar tools supporting SiGML. It performs a wide range of manual gestures (hand movements, finger shapes) and non-manual expressions (facial expressions, head movements), ensuring natural and accurate translation.

Real-Time Animation: The system processes SiGML files dynamically, enabling the avatar to respond to each word or phrase in real-time, providing instant visual feedback and translating spoken Bangla into BSL gestures seamlessly.

Web-Based Implementation: The avatar runs on a web-based platform. This will be accessible across devices such as computers and smartphones. This makes it a flexible, and convenient solution for real-time communication. This stage enhances accessibility for the hearing-impaired.

3.3.4 Key Performance Indicators for Voice to Sign Generation

Our research work is divided into two key performance evaluation methods. The first one is Bangla Voice-to-Word Conversion, and the second one is the 3D Animation Avatar system. These methods are designed to comprehensively assess the accuracy and effectiveness. Also, we have evaluated the real-time performance of the proposed system for translating Bangla voice into BDSL using 3D avatars.

(i) Bangla Voice-to-Word Conversion Performance Evaluation

The first method focuses on the accuracy and efficiency of converting Bangla voice input into Bangla text. This step serves as the foundation for the sign language translation. The performance metrics for this stage include below:

Speech-to-Text Accuracy: It measures how accurately the system converts spoken Bangla into written text. The system must accurately transcribe various dialects and accents of Bangla while maintaining a high level of precision. The accuracy rate mentions the percentage of correctly transcribed words, which is then compared to the total number of words spoken.

$$Accuracy = \frac{Correctly\ Transcribed\ Words}{Total\ Words\ Spoken} \times 100 \quad (3.1)$$

Real-Time Processing Speed: This metric evaluates the time it takes for the system to convert voice input into text in real-time. Speed is critical in real-time applications. Live conversations

or sign language interpretation. It ensures that the average processing time per word is under 1.6 seconds, allowing for real-time interaction.

$$Processing\ Speed = \frac{Total\ Processing\ Time}{Number\ of\ Words} \tag{3.2}$$

(ii) 3D Animation Avatar Performance Evaluation Formula

To evaluate the quality and effectiveness of the sign animations, we have conducted the evaluation by seven professional Bangla Sign Language interpreters. We have assessed various animations across categories such as alphabets, numbers, sentences, and words. The evaluation has been done based on a five-point rating scale. The following categories: Poor, Fair, Average, Good, and Excellent. Each rating will be assigned a corresponding numeric score of 1, 2, 3, 4, or 5, respectively. The overall performance score will be calculated using the formula:

$$Score = \frac{P \times 1 + F \times 2 + A \times 3 + G \times 4 + E \times 5}{N} \tag{3.4}$$

where:

- P = Number of "Poor" ratings
- F = Number of "Fair" ratings
- A = Number of "Average" ratings
- G = Number of "Good" ratings
- E = Number of "Excellent" ratings
- N = Total number of ratings

Performance Calculation for Different Sections:

Alphabets Section: The interpreter will assess 26 alphabet animations. The ratings from this section will be used to calculate the average score. This reflects the clarity and accuracy of the alphabet animations.

Numbers Section: In this section, animations representing numerical symbols from 0 to 9 have been evaluated. High scores in this section will indicate satisfaction with how numerical gestures are represented through avatar animations.

Words Section: A comprehensive evaluation of word animations will be conducted. This section provides insights into how well the system translates individual words into Bangla Sign Language gestures. We have shown the evaluation for 135 words.

Sentences Section: The interpreter will evaluate the animations in sentences. The performance score in this section has been evaluated on the increased effectiveness of a system with longer. We have included more complex gestures that are combinations of multiple signs.

Overall Performance Score: By aggregating the scores of all the sections, we have considered the overall average score. This score will offer a broad form of the system's effectiveness and overall user satisfaction with the quality of the animations of the signs.

(iii) Performance Evaluation Process

Alphabets: We have chosen the English alphabet (A to Z) of 26 letters instead of the Bangla alphabet. The reason for this choice is that the English alphabet is more in use for communicating by sign language. For spelling out words, names, and technical terms that may not have direct signs in Bangla. In order to assess the quality and accuracy of our sign language animations, we prepared and made the animation available via Google Form. Seven sign language experts in total took part in this evaluation process. Each of the experts worked through the animations carefully for all 26 letters of the alphabet and issued marks according to what they had seen. The evaluation was conducted on a five-point Likert scale in which:

- Poor - indicates very poor or inaccurate animation
- Excellent - indicates excellent and highly accurate animation.

Thus, for each alphabet, seven scores were collected one from each expert. These ratings were then compiled and analyzed to measure the overall accuracy, consistency, and quality of the animations. This systematic evaluation helped us understand not only the strengths of our animated signs but also the areas that require improvement.

Figure 3.13: Google Form for Alphabet Evaluation screenshot for voice-to-sign animation.

Numbers: In addition to alphabets, we have also focused on the numerical digits from 0 to 9. These ten digits were chosen because numbers play a vital role in everyday communication through sign language, such as expressing dates, phone numbers, time, money, and other quantitative information. Unlike alphabets, which are mostly used for spelling, numbers are frequently used as standalone signs, making their accurate representation equally important.

To evaluate the quality and correctness of our sign language number animations, we shared the animations with a group of seven sign language experts using a Google Form. Each expert carefully observed the animations for digits 0–9 and rated them according to their clarity, accuracy, and naturalness.
The evaluation used the same five-point Likert scale:

- Poor - indicates very poor or inaccurate animation,
- Excellent - indicates excellent and highly accurate animation.

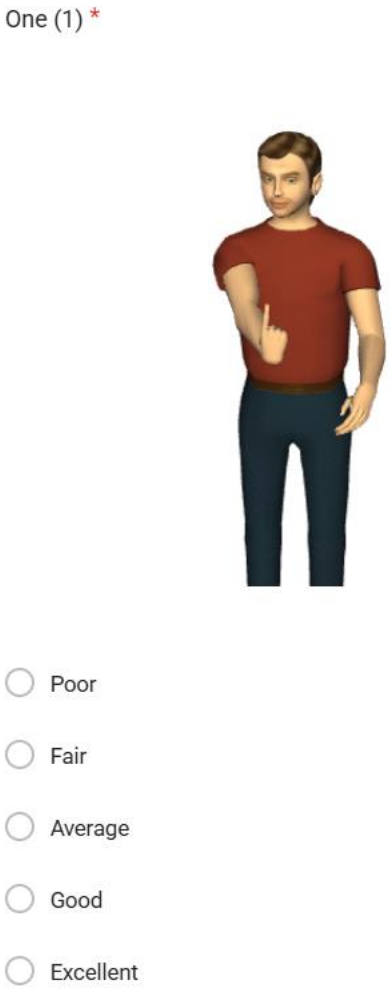


Figure 3.14: Google Form for Numbers Evaluation screenshot for voice-to-sign animation.

As a result, ten separate ratings were collected from each expert, corresponding to the ten digits. These evaluations were then compiled to measure the average accuracy, reliability, and

acceptance of the number animations. This approach ensured that the developed number signs were systematically tested and validated by experienced sign language practitioners.

Words: Alongside alphabets and numbers, we have also considered a set of commonly used words in daily communication. These words were selected because they represent frequent interactions in sign language, such as greetings, questions, and essential expressions. Unlike alphabets, which are primarily used for spelling, and numbers, which are used for quantitative information, words carry direct semantic meaning and therefore require precise and natural animation to ensure effective communication. To evaluate the word-level sign animations, we shared our animations through a Google Form with seven experienced sign language experts. Each expert carefully reviewed the animations for every selected word and assigned a score based on clarity, accuracy, and natural flow.

The evaluation followed the same five-point Likert scale:

- Poor - indicates very poor or inaccurate animation,
- Excellent - indicates excellent and highly accurate animation.

Thus, for each word, seven ratings were collected one from each expert. These scores were then analyzed to determine the average performance, accuracy level, and overall acceptance of the word-based animations. This evaluation process helped us identify how well our system performs in producing contextual and meaningful signs, compared to alphabets and numbers.

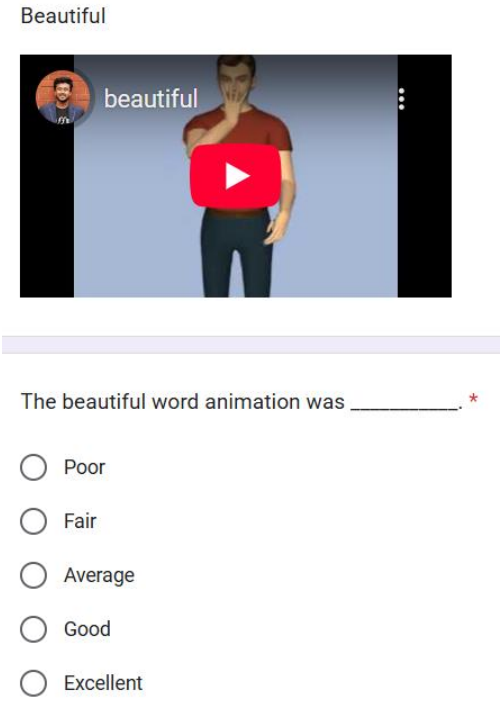


Figure 3.15: Words Evaluation screenshot for voice-to-sign animation.

Sentences: In addition to alphabets, numbers, and words, we have also worked with complete sentences. Sentences were chosen because they represent real-life communication scenarios,

where signers often express full ideas rather than isolated letters or words. Animating sentences is more complex, as it requires not only accurate signs for individual words but also maintaining natural flow, grammar, and continuity in sign language.
To evaluate the quality of the sentence-level sign animations, we shared the animations through a Google Form with seven sign language experts. Each expert carefully reviewed the animations of the selected sentences and rated them based on fluency, accuracy, clarity, and naturalness of expression.
The evaluation used the same five-point Likert scale:

- Poor - indicates very poor or inaccurate animation,
- Excellent- indicates excellent and highly accurate animation.

For each sentence, seven ratings were collected from the experts. These scores were compiled and analyzed to measure the overall accuracy, fluency, and effectiveness of our sentence-level sign animations. This evaluation allowed us to assess how well our system performs in generating natural, meaningful communication, moving beyond isolated letters and words.

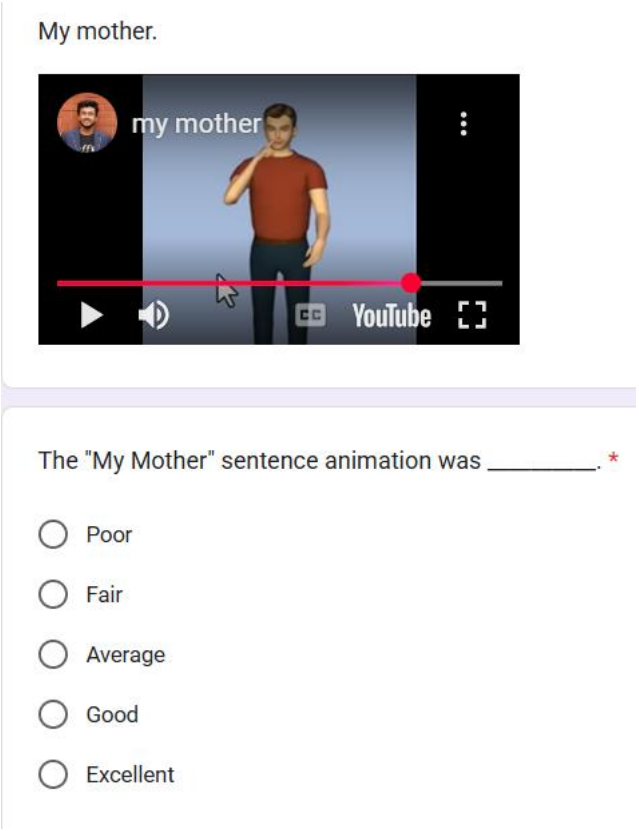


Figure 3.16: Sentences Evaluation screenshot for voice-to-sign animation.

3.4 SIGN LANGUAGE TO VOICE GENERATION

Conversely, the system can interpret sign language gestures made by a deaf individual, converting these into both text and spoken Bangla. This reverse communication process is depicted in Figure 3.13.

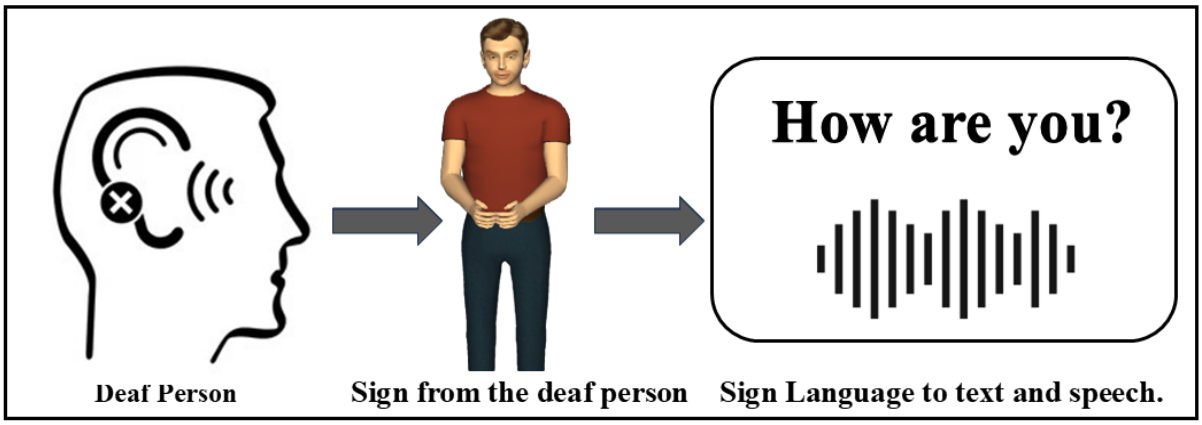


Figure 3.17: Sign Language to Voice and Text Conversion Process.

The diagram below shows a video-based sign language recognition process. It starts with video input, which is broken down into individual frames. These frames undergo image preprocessing involving enhancement, resizing, and data augmentation to improve quality and consistency.

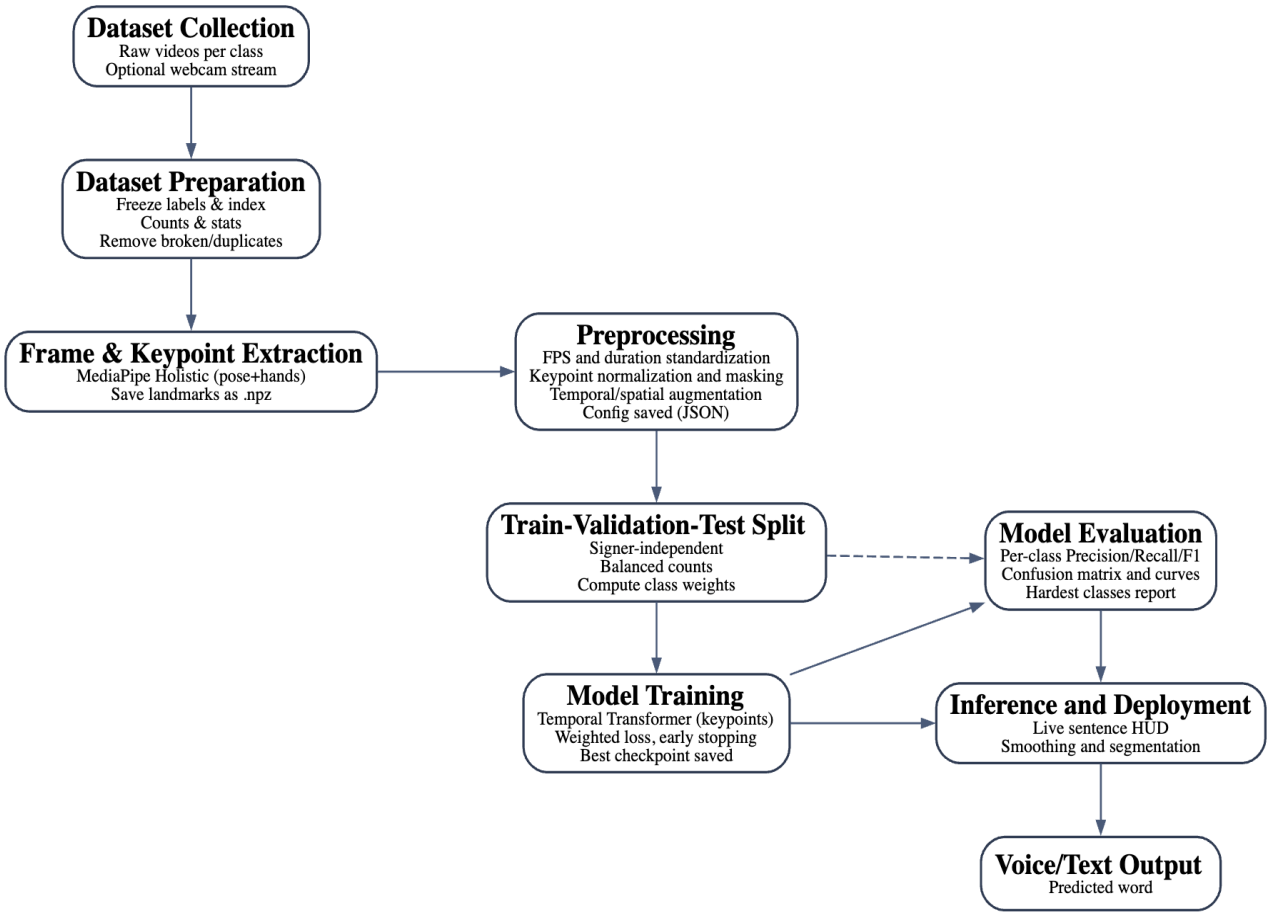


Figure 3.18: Workflow diagram for sign language to voice and text conversion.

3.4.1 Dataset Collection

Data collection is the first step for sign language to voice and text conversion. The figure below illustrates is a comprehensive dataset collection process designed to enhance the robustness and accuracy of a sign language recognition system.

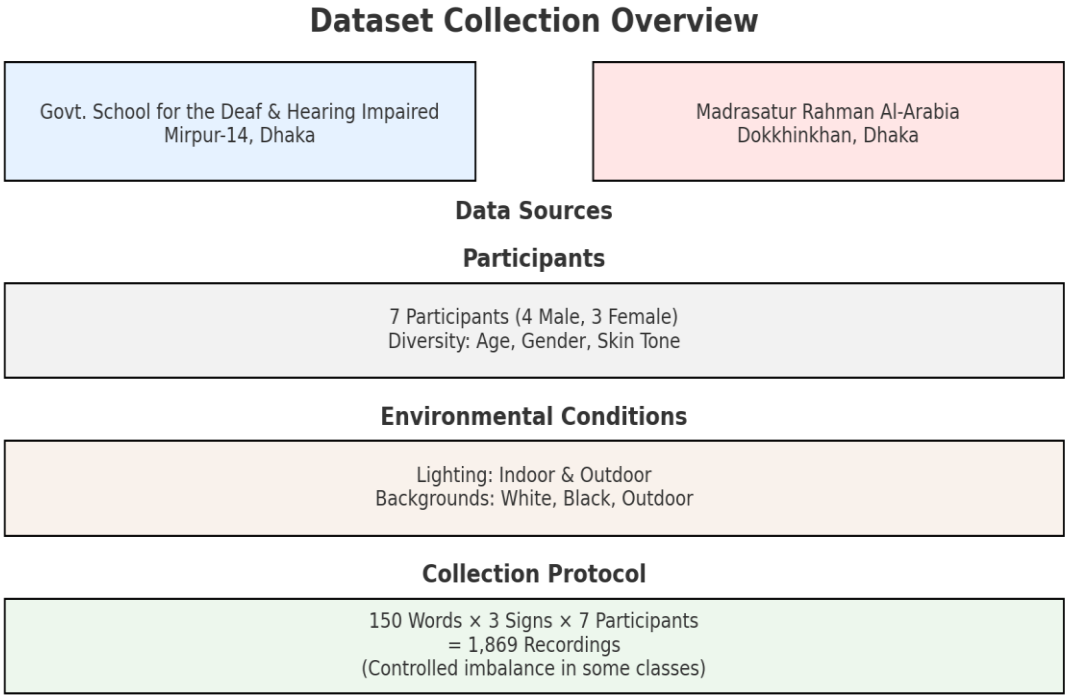


Figure 3.19: Dataset collection from 7 different signers.

To ensure comprehensive coverage of sign language vocabulary for effective communication, data were collected for 150 distinct words. This vocabulary size was chosen to balance both breadth and feasibility, providing sufficient linguistic variety to train and evaluate the recognition system.

Sources of Data Collection: Data were gathered from two different institutions in Dhaka, Bangladesh, with the aim of incorporating participant diversity and contextual variation:

i) Government School for the Deaf & Hearing Impaired (Mirpur-14, Dhaka)

This institution is a specialized school dedicated to the education and development of deaf and hearing-impaired students. Data collection was carried out with the support of four experienced sign language teachers:

- Badrunnesa Bithi
- Umme Roman Siddika
- Amena Akter
- Towhidul Islam

ii) Madrasatur Rahman Al-Arabia (Dokkhinkhan, Dhaka)

This institute also provides education for students with hearing impairments. Data were collected from sign language teachers of the deaf department, specifically:

- Rezuan Saleh
- Atahar Ali
- Jahid Hasan

In total, seven participants contributed to the dataset, comprising four males and three females. The group represented varying ages and skin tones, thereby ensuring demographic diversity. This demographic variation was intentionally incorporated to improve the generalizability of the model across different individual characteristics. To enhance robustness, the dataset was collected under controlled variations in environmental conditions:

Age Diversity: We have collect data from different age people and there age is from 25 years to 52 years. There cloth are also different and some cloths are covered their elbow. These are are the diversity in our dataset.

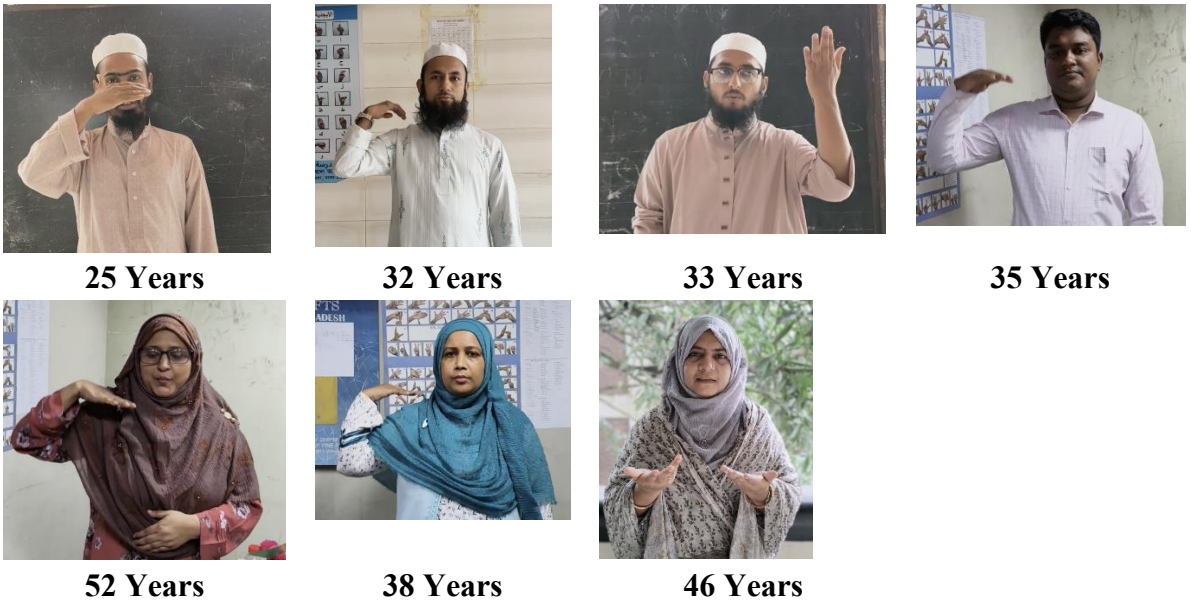


Figure 3.20: All signers sample data along with their age.

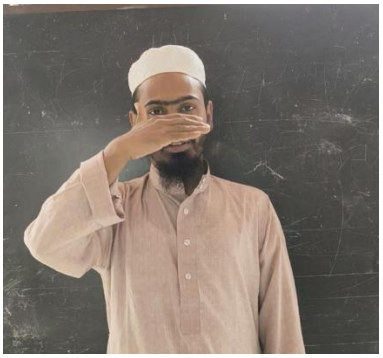
Lighting conditions: Both indoor and outdoor illumination settings were included to enable the system to adapt to different light intensities and directions.

Low light

High Light

Figure 3.21 : Dataset sample for low and high light.

Background conditions: Three distinct background settings were applied such as white, black, and outdoor environments. To capture potential variations in contrast and background complexity.



Black Background

Indoor Background

Outdoor Background

Figure 3.22: Dataset sample for different backgrounds.

Device Diversity: To further enhance the dataset's robustness and enable the sign recognition model to generalize effectively across diverse hardware, we employed three distinct mobile devices with varying camera configurations:

- Google Pixel 7: Featuring a high-resolution 50 MP rear wide-angle lens (f/1.85, ~82° field of view, 1.2 µm pixels).
- POCO X3: A mid-range alternative equipped with a 64 MP main sensor (Sony IMX682, f/1.89, 1.6 µm).
- OnePlus Nord: Balancing quality and affordability, this device includes a 48 MP main sensor (Sony IMX586, f/1.75).

Each participant was instructed to perform all 135 words, with three distinct sign variations per word. Signs were recorded from different angles, introducing spatial diversity into the dataset. This protocol resulted in 405 recordings per participant (135 words × 3 signs). With seven participants, the total dataset size reached 1,869 recordings. This large and diverse corpus means that the system is exposed to a great diversity of signing styles, physical characteristics,

and environmental conditions. The data is from a demographically diverse population. It has controlled unbalanced in some word classes. This is to test the robustness of the model for multiple distributions. The data are for a variety of lighting and background conditions. These have guaranteed flexibility for real-life situations.

This rich dataset is very important to train the sign language recognition model, which includes linguistic diversity, diversity of participants, and environmental heterogeneity. It forms a proper basis on which to build up the knowledge required to develop a system that can interpret BSL accurately even in the various real-world conditions.

Dataset Statistics and Characteristics

A proper analysis of the dataset is crucial to ensure the validity of the dataset in building a robust Bangla Sign Language recognition system. It is important to understand the distribution of the data in the classes. The participants are represented, and the temporal properties of the videos vary. These statistics give us insight into the balance, diversity, and representativeness of the dataset. It ensures that it reflects the conditions that the recognition model will operate in the real world. Here, we provide an elaborate look at the dataset, accompanied by some visualizations to emphasize the class distributions and participant diversity and variability in duration.

Duration per Class: Figure 3.3 presents the distribution of video durations across the top 20 classes in the dataset. The boxplots demonstrate that the majority of videos fall within 2 to 3 seconds. There are some videos extending up to 5 seconds. Variability is evident across classes, where certain signs. For example, “Boro Vai” and “Dada” show a wider spread of durations. Some other words, like “Bon” or “Amar”, remain more consistent. This natural variation reflects differences in individual signing speed and style. Such diversity is beneficial for training models that must generalize to real-world usage, where users may perform signs at different speeds and with varying emphasis.

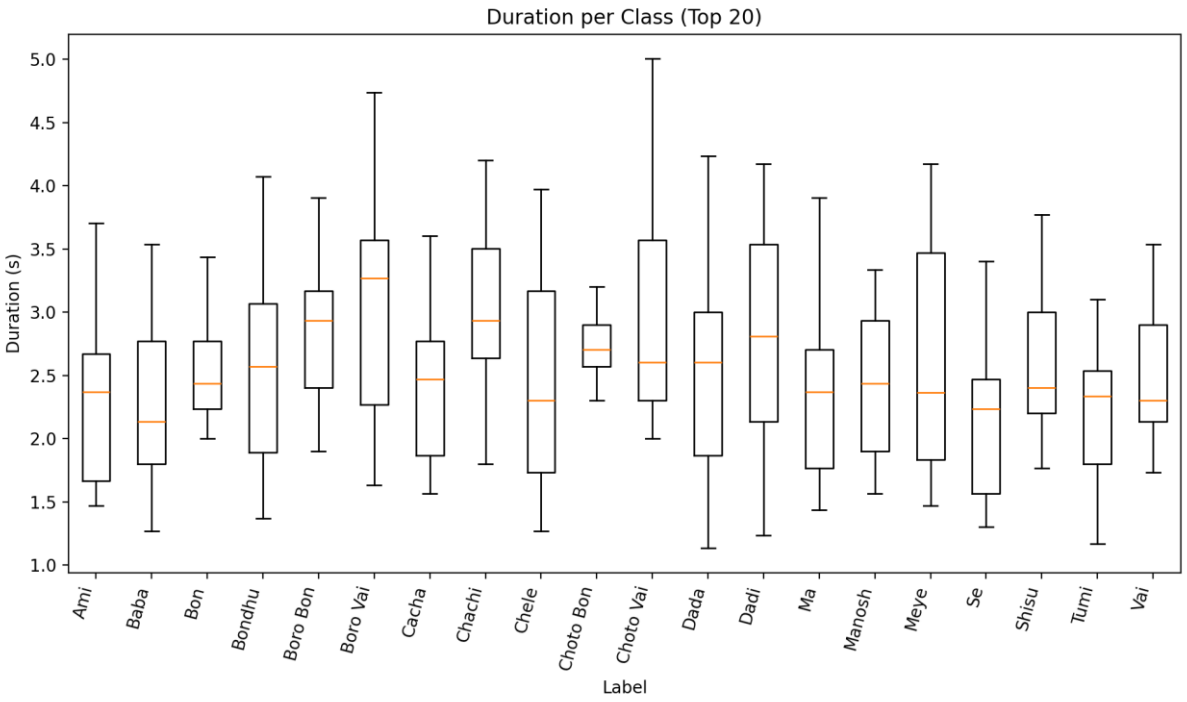


Figure 3.23: Distribution of video durations across the top 20 classes.

The presence of both long and short duration classes ensures that the dataset captures the temporal diversity of BSL. This variety helps in training models that can adapt to different word complexities, signing speeds, and gesture lengths.

Table 3.4: Duration Statistics for Longest and Shortest Classes.

Category	Class	Mean Duration (s)	Median Duration (s)
Longest Duration	Debor	3.76	3.20
	Khalo	3.73	3.74
	Fufa	3.65	3.74
	Shashuri	3.64	3.40
	Konna	3.51	3.47
Shortest Duration	Ek	1.43	1.80
	Choy	1.49	1.60
	Doi	1.67	1.89
	Boro	1.68	1.50
	Baire	1.69	1.70

Participant Diversity: Figure 3.21 shows a grid of keyframes sampled from the recordings of the seven participants. The dataset includes four male and three female signers.

Figure 3.24: Grid of keyframes sampled from the recordings of the seven participants.

Participants also differ in age, clothing, and appearance, providing further demographic diversity. These frames highlight recordings performed under varied environmental conditions, including both indoor classrooms and outdoor scenes. We have also added different lighting and background settings. This diversity is critical for minimizing dataset bias and ensuring that the trained recognition system is capable of handling the variability of real-world users.

Class Distribution: The distribution of the video samples within all the word classes is shown in Figure 3.5. The dataset preserves a fairly balanced structure with the majority of classes having between 12 and 21 videos. Some classes are deliberately underrepresented, so that a controlled class imbalance is created. This type of design choice enables researchers to test how models for recognition work under realistic conditions. Some signs are more common than others. Overall, the distribution is guaranteed to have enough representation of all 135 word classes.

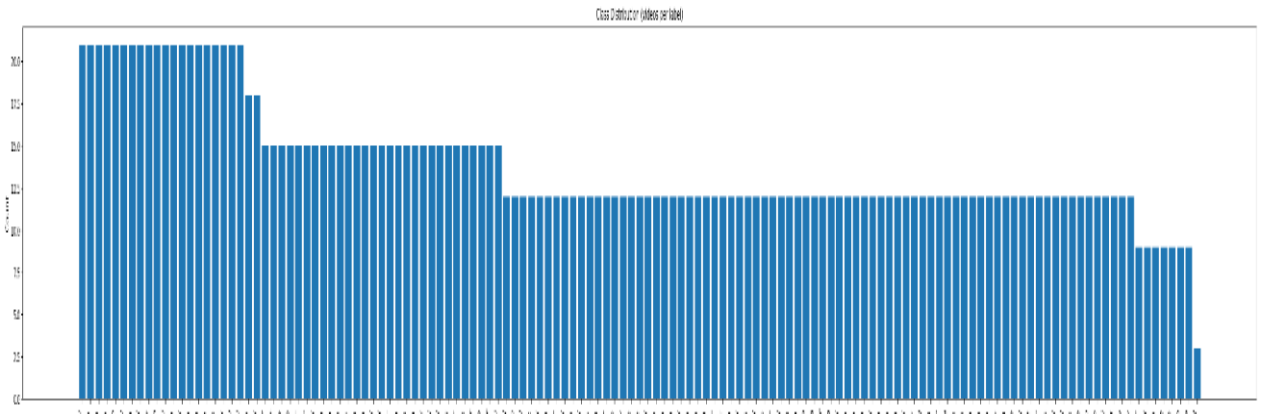


Figure 3.25: The distribution of video samples for all word classes.

3.4.2 Dataset Preparation

The Dataset Preparation stage involves organizing and structuring the raw data. We need to organize in a format that is suitable for machine learning tasks. This step is done to make the dataset uniform and clear. This prepares it for training the model. The major tasks in this step are to freeze the labels, to create index files, and to eliminate corrupted or duplicated files from the dataset. These activities are used to normalize the dataset. These techniques provide an efficient way of training and evaluating the model.

i) Freeze Labels & Index

In this phase, the label of the dataset is frozen to achieve consistency in training the model. This is done by generating an index of all labels and giving a unique ID to each label. A simple text file that contains all the labels, one per line, sorted alphabetically. A JSON file that contains the labels in a list format. A JSON file mapping each label to a unique integer ID. These files help the model understand which label corresponds to which category during training and evaluation.

ii) Counts & Stats

The script also collects statistics about the dataset. It counts how many files (videos) exist for each label (class) and stores this information some files. A file that contains a file-level index with details such as the path, label, extension, and size of each vide. Another file containing the count of video files for each label, which helps in understanding the class distribution and can be used for balancing the dataset. These steps ensure that all the necessary metadata is available for dataset analysis, preprocessing, and model training.

iii) Remove Broken/Duplicates

The script does not specifically mention broken file removal in the code, but it ensures that only valid files (those with a recognized video extension) are included in the dataset. If there are any corrupted or inaccessible files, they will be ignored during the indexing process. In case of

duplicates, the file path is checked, and if the same file exists, it is skipped during the process.

3.4.3 Frame and Keypoint Extraction

The Frame & Keypoint Extraction step is a crucial part of the data processing pipeline where keypoints are extracted from videos using MediaPipe Holistic. This process involves extracting pose and hand landmarks (total of 75 keypoints) from each frame of the video, which are essential for sign language recognition.

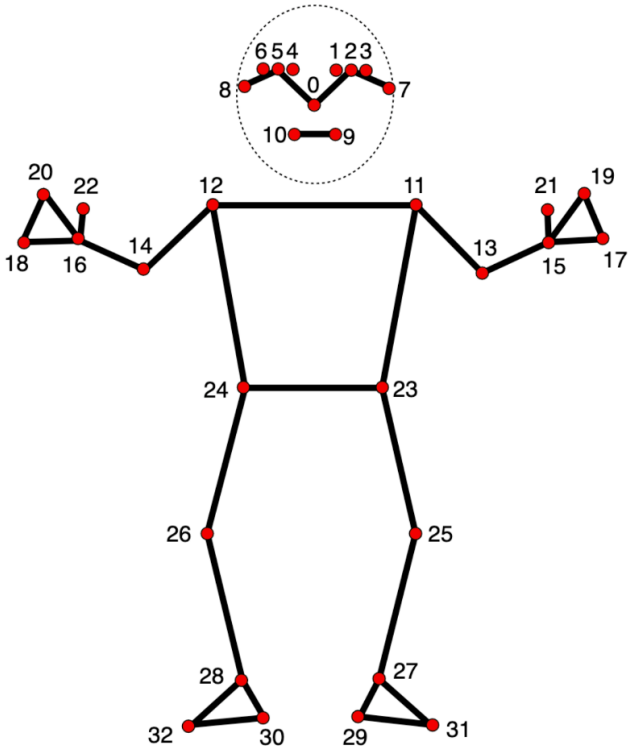


Figure 3.26: The pose landmarker model tracks 54 body landmark locations.

The output of this step is a sequence of 3D coordinates for each keypoint, along with their visibility information, saved in a compressed .npz file format for efficient storage and retrieval. In this step, we use MediaPipe Holistic, a state-of-the-art tool developed by Google, to perform real-time detection of pose landmarks (33 keypoints) and hand landmarks (21 keypoints per hand). The keypoints are then normalized and scaled to ensure consistency across different gestures and video lengths. These normalized coordinates are saved as part of a sequence for each video clip.The pose landmarker model tracks 33 body landmark locations, representing the approximate location of the following body parts:

Table 3.5: Name of each landmark.

0 - nose	11 - left shoulder	22 - right thumb
1 - left eye (inner)	12 - right shoulder	23 - left hip
2 - left eye	13 - left elbow	24 - right hip
3 - left eye (outer)	14 - right elbow	25 - left knee
4 - right eye (inner)	15 - left wrist	26 - right knee

5 - right eye	16 - right wrist	27 - left ankle
6 - right eye (outer)	17 - left pinky	28 - right ankle
7 - left ear	18 - right pinky	29 - left heel
8 - right ear	19 - left index	30 - right heel
9 - mouth (left)	20 - right index	31 - left foot index
10 - mouth (right)	21 - left thumb	32 - right foot index

i) MediaPipe Holistic (pose + hands):
MediaPipe Holistic is used for detecting the full set of landmarks required for hand and pose recognition. This includes:

- Pose Landmarks: 33 keypoints that define the skeletal structure of the body.
- Left and Right Hand Landmarks: 21 keypoints for each hand representing finger joints and palm orientation.
- The MediaPipe Holistic model is configured with certain parameters:
- Model Complexity: Set to 1 for a balance between performance and accuracy.
- Static Image Mode: Set to False, enabling dynamic video processing.
- Smoothing Landmarks: Applied to smooth the detected keypoints over time to reduce jitter.

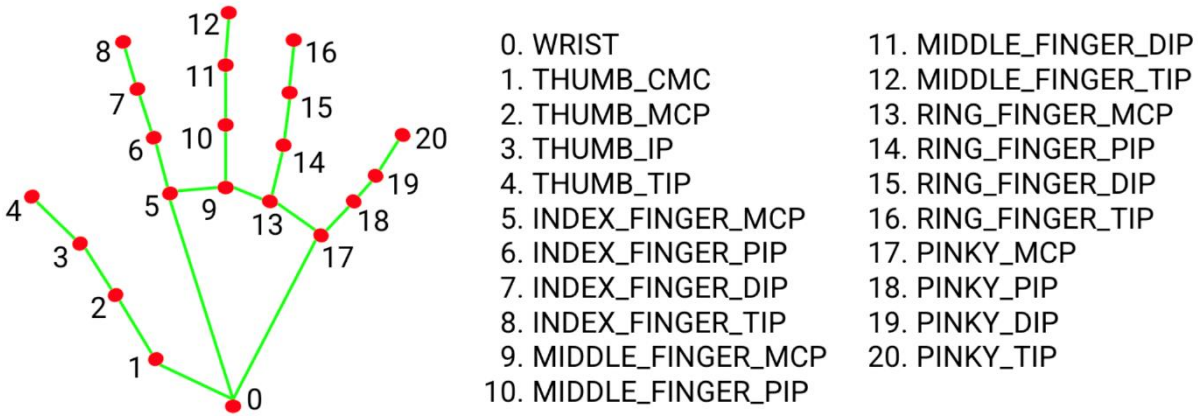


Figure 3.27: Hand landmarks and their names.

The part captures each video frame, processes it to extract the required landmarks, and scales the keypoints for consistent measurements. The results for each frame are stored as sequences, with each frame containing 75 keypoints in 3D space (x, y, z), along with visibility data that indicates the confidence of each landmark being visible in the frame. A sample image for mediapipe landmarks is showen in figure 3.23.

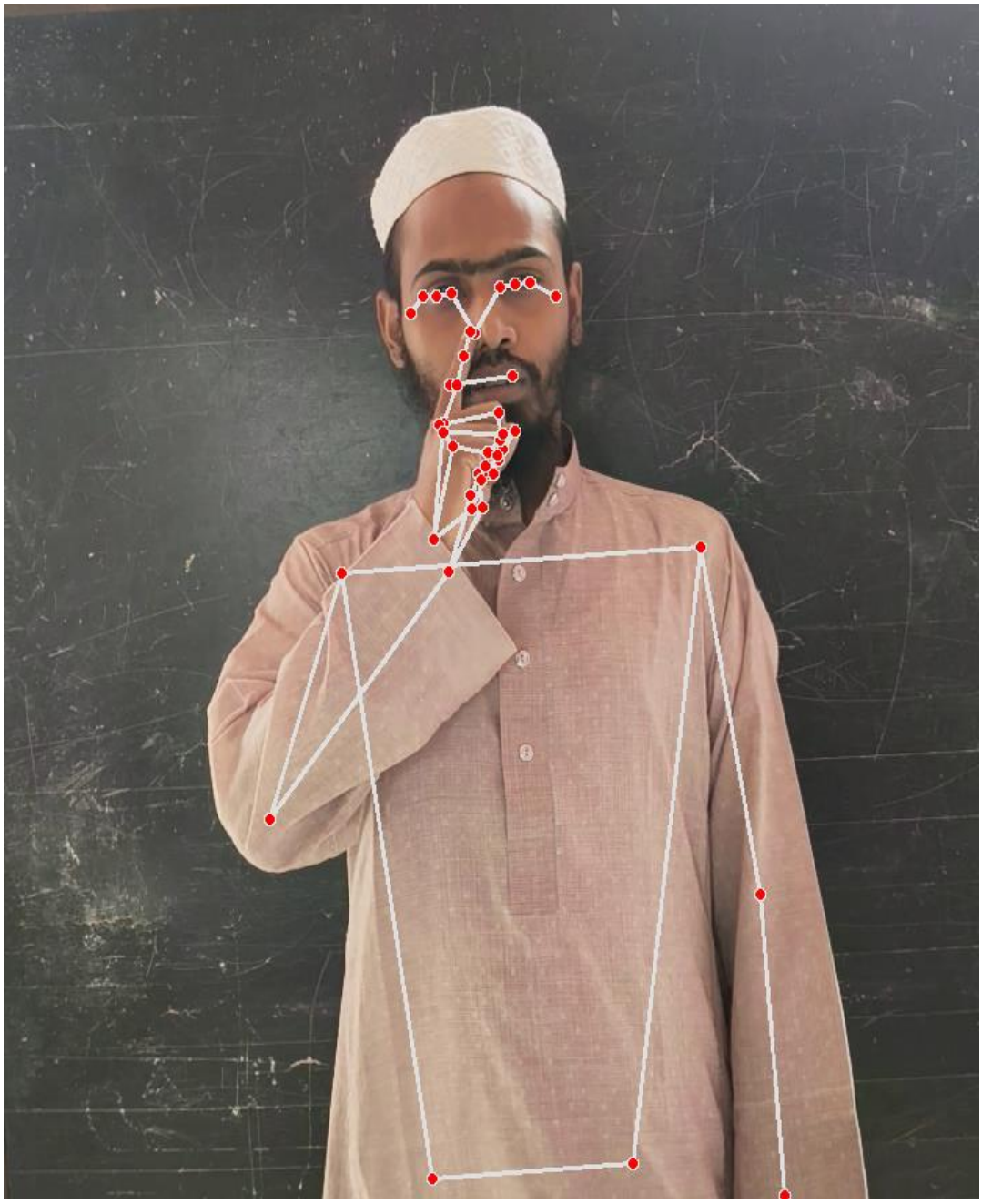


Figure 3.28: Sample image for Mediapipe landmark for the word “Ma”.

Saving Landmarks as .npz:
The keypoints and visibility values are saved in .npz format for each video. The x array contains the 3D coordinates for all keypoints over time, with dimensions [T, 75, 3], where T is the number of frames in the video. The vis array holds the visibility information for each keypoint, with dimensions [T, 75]. The metadata about each video is also stored in a JSON format and included in the .npz file, which contains details such as the label, split (train/val/test), and normalization method used for the keypoints.

3.4.4 Dataset Preprocessing

Preprocessing of the raw video data is a crucial step in ensuring consistent and reliable inputs for the word-level Bangla Sign Language recognition model. In this study, preprocessing involved standardizing video frame rates, normalizing keypoints, applying temporal and spatial augmentations, and saving a frozen configuration for reproducibility. Each step is described in detail below.

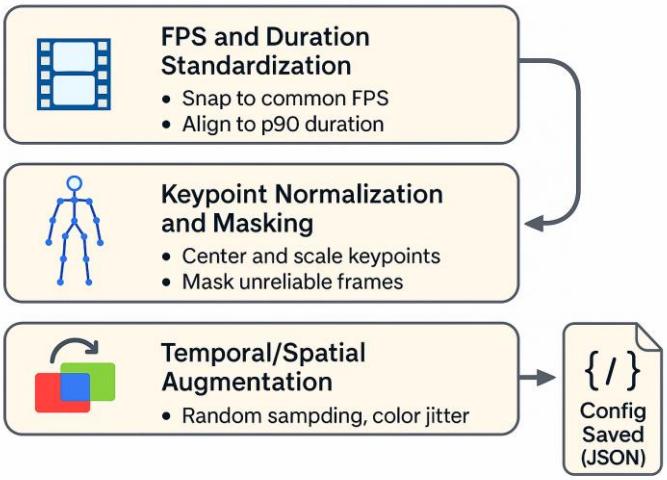


Figure 3.30: Preprocessing flow diagram.

FPS and Duration Standardization: Sign language videos in the dataset were captured using different devices, leading to variations in frame rates (FPS) and video durations. To ensure temporal consistency across all samples, the preprocessing pipeline automatically derived the most common FPS value from the metadata and aligned all videos to the nearest standard FPS (e.g., 24, 25, 29.97, 30, 50, 60 fps). This approach reduces temporal discrepancies that could negatively affect model learning. Additionally, the duration of video clips was standardized using the 90th percentile (p90) of the recorded durations, ensuring that the majority of videos were fully represented while ignoring extreme outliers. A minimum duration of 4 seconds was enforced to retain sufficient temporal information. The number of frames per clip (FRAMES_PER_CLIP) was then calculated as:

$$\text{FRAMES PER CLIP} = \text{TARGET FPS} \times \text{CLIP SECONDS} \tag{3.5}$$

This ensures that every video input fed to the model has a consistent temporal length.

Keypoint Normalization and Masking: For skeleton-based recognition, each video frame was processed using MediaPipe Holistic to extract 75 keypoints per frame (33 for the full body pose, 21 for each hand). To remove inter-signer variation and spatial inconsistencies, keypoints were normalized based on shoulder distance and centered around the torso. Frames that could not be reliably processed or had missing keypoints were automatically masked to prevent propagating noise into the model. This normalization ensures that the learning model focuses on the relative motion of joints rather than absolute position differences, making it more robust to variations

in signer height, camera angle, and distance from the camera.

Temporal and Spatial Augmentation: To increase the generalizability of the model, several data augmentations were applied:

- Temporal augmentation: Random frame sampling and slight speed perturbations simulate variations in signing speed.
- Spatial augmentation: Resize strategy: All frames were resized to a target size of 224 × 224 pixels while maintaining aspect ratio using center crop or padding.

These augmentations help the model learn robust features across different lighting conditions, signer orientations, and minor temporal inconsistencies.

Config Saving (JSON and YAML): To ensure reproducibility and facilitate future experiments, the preprocessing targets and augmentation parameters were frozen in a configuration file and saved in both YAML and JSON formats. The configuration file contains the following information:

- Target FPS and resolution
- Clip duration and number of frames per clip
- Skeleton processing details (keypoints per frame, normalization strategy)
- Metadata-derived modes for FPS, resolution, and duration
- Source paths and project information

This configuration allows the preprocessing pipeline to be re-run consistently at any point, ensuring that extracted features and subsequent model training are fully reproducible.

Table 3.6: Summary of Preprocessing Configuration.

Parameter	Value
Target FPS	30
Target Resolution	224 × 224 (keep aspect ratio + center crop/pad)
Clip Duration	4–p90 seconds
Frames per Clip	Target FPS × Clip Duration
Skeleton Joints per Frame	75

Keypoint Normalization	Centered & scaled by shoulder distance
Spatial Augmentation	Color jitter ±10%, horizontal flip 0.5
Config Saved	YAML & JSON

This preprocessing pipeline ensures temporal consistency, spatial normalization, and robustness to variations in signer appearance and recording conditions, providing a standardized input for both skeleton-based models. The frozen configuration allows future researchers to reproduce the preprocessing exactly as performed in this study.

3.4.5 Train-Validation-Test Split

To ensure fair evaluation and strong generalization, the dataset was divided into train, validation, and test sets using a structured splitting strategy. Whenever possible, the split was performed in a signer-independent manner, meaning that the same signer did not appear in both training and evaluation splits for the same class.

Signer-Independent Splitting: To prevent the model from memorizing signer-specific features such as skin tone, body proportions, or signing style, a signer-independent strategy was employed wherever possible. This was achieved by parsing signer IDs from filenames using a regular expression. If a class had at least two or more unique signers, its data was divided so that clips from the same signer appeared only in one split (train, validation, or test). This ensures that validation and test accuracy reflect the model's ability to generalize to new signers, not just to previously seen individuals. In cases where only a single signer existed for a class, the algorithm fell back to a clip-level stratified split, preserving balance while ensuring reproducibility with a fixed SEED = 42.

Balanced Counts Across Splits: The dataset was divided into 80% training, 10% validation, and 10% testing, as specified by:

$$RATIOS=\{train: 0.80, val: 0.10, test: 0.10\}$$

To achieve this balance, we have processed each class independently. If a class had fewer than 3 samples, all clips were placed into the training set to prevent degenerate splits. Otherwise, the algorithm attempted signer-based splitting first, and clip-based splitting second (if signers were not available). After the initial assignment, a sha1-cohesion check ensured that identical or duplicate clips (detected by identical hash values) were not split across train/val/test. Instead, all duplicates were moved into the majority split, preventing data leakage. This process yielded deterministic and reproducible splits, ensuring consistency across repeated experiments.

Table 3.7: Train-validation-test Split Statistics.

Split	Number of Clips	Percentage
Train	1,495	80%
Validation	187	10%
Test	187	10%
Total	1,869	100%

Table 3.8: Class weights summary for some random class.

Class	Training Clips	Class Weight
Ek	8	186.9 / 8 = 23.36
Debor	21	186.9 / 21 = 8.9
Khalo	18	186.9 / 18 = 10.38
Boro	12	186.9 / 12 = 15.58
Amar	21	186.9 / 21 = 8.90

Here 186.9 ≈ total train samples normalized; values shown for illustration, actual values come from the training set counts.)

3.4.6 Model Training

The Temporal Transformer model was employed to handle the recognition and classification tasks based on keypoints extracted from video frames. The Transformer architecture is especially well-suited for tasks. This involves sequential data because of its ability to capture long-range dependencies without the need for RNNs or CNNs. The self-attention mechanisms are used in the model. This enables it to pay attention to relevant sections of the input sequence. This makes it particularly powerful for tasks involving spatial-temporal data, such as sign language recognition.

The model architecture is based on the original Transformer architecture, which was introduced by Vaswani et al. (2017) in “Attention is All You Need”[55]. The Transformer is an encoder-decoder model. The encoder takes the input sequence and passes it through several layers of multi-head attention and feedforward layers. This architecture is suitable for sequential data and has achieved the state-of-the-art performance in several domains. These are machine translation, and image processing.

i) Model Architecture: The Transformer consists of an encoder and decoder architecture with the following key components:

- Encoder: Composed of stacked layers where each layer consists of:
  - Multi-head self-attention mechanism.
  - Position-wise feed-forward network.
  - Each layer has a residual connection followed by layer normalization.
- Decoder: Similar to the encoder, but with an additional attention mechanism that attends to the encoder's output.

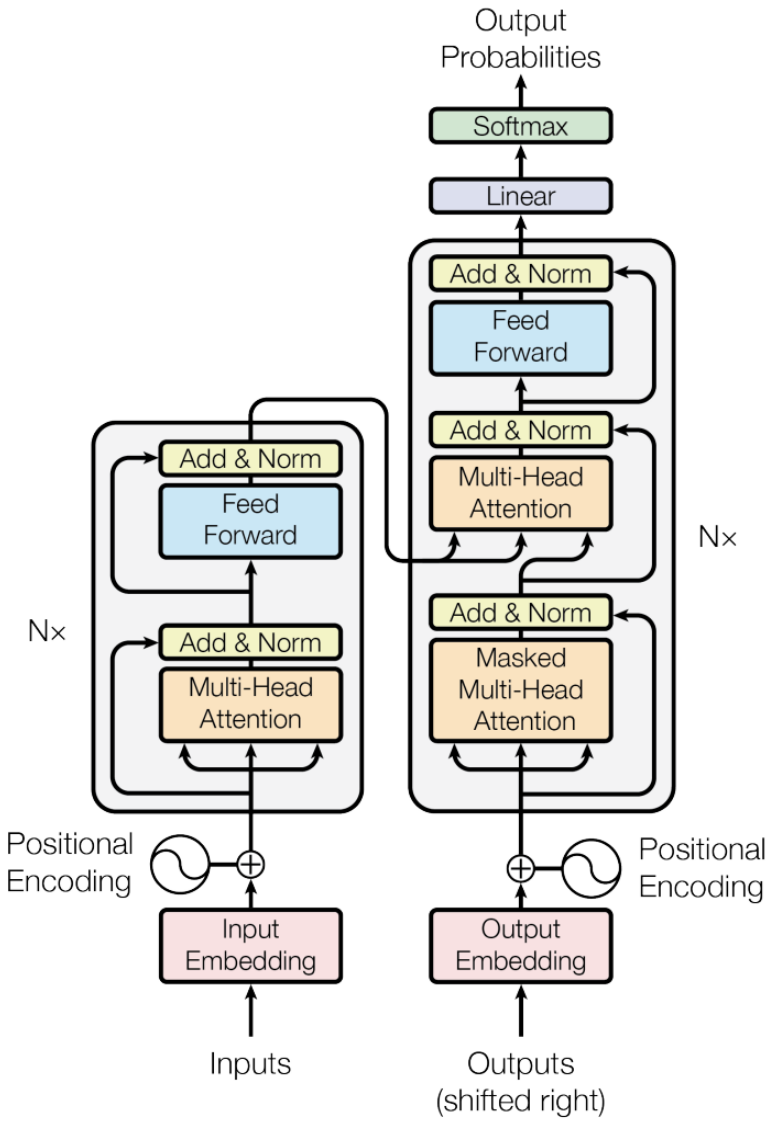


Figure 3.31: Here is a simplified depiction of the model architecture.

ii. Equations: The model works by using the following four equations-

1. Scaled Dot-Product Attention
The attention mechanism is defined as:

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V \tag{3.6}$$

Where:

Q = Query matrix
K = Key matrix,
V = Value matrix,
d_k = Dimensionality of the key

2. Multi-Head Attention:

The multi-head attention is computed as:

$$\text{MultiHead}(Q, K, V) = \text{Concat}(head_1, ..., head_h)W_O \tag{3.7}$$

where each $head_i$ is computed as:

$$head_i = \text{Attention}(QW_Q^i, KW_K^i, VW_V^i) \tag{3.8}$$

and W_Q^i, W_K^i, W_V^i are learned projections.

3. Position-wise Feed-Forward Networks:

Each layer in the encoder and decoder contains a fully connected feed-forward network defined as:

$$\text{FFN}(x) = \max(0, xW_1 + b_1)W_2 + b_2 \tag{3.9}$$

where W_1, W_2 are weight matrices, and b_1, b_2 are bias terms.

4. Positional Encoding:

The positional encoding is added to input embeddings to provide information about the order of the tokens:

$$PE(pos, 2i) = \sin\left(\frac{pos}{10000^{2i/d_{\text{model}}}}\right) \tag{3.10}$$

$$PE(pos, 2i+1) = \cos\left(\frac{pos}{10000^{2i/d_{\text{model}}}}\right) \tag{3.11}$$

where pos is the position, and i is the dimension index.

iii. Attention Mechanisms

The model uses three types of attention:

- Self-Attention: In this mechanism, each token in a sequence attends to every other token, capturing long-range dependencies.

- Encoder-Decoder Attention: The decoder attends to all positions in the encoder's output.
- Masked Self-Attention in Decoder: Ensures that predictions for position iii only depend on positions less than iii, maintaining the autoregressive property.

The training process of the Transformer model is iterative, consisting of multiple epochs, where the model continuously updates its internal parameters (weights and biases) to minimize the error in its predictions. In each epoch, the model is presented with a batch of input data, and for

each input, the model generates predictions based on the current state of its parameters. These predictions are compared with the actual labels (ground truth) to compute the error or loss. The error is typically calculated using a loss function, such as categorical cross-entropy for classification tasks. This loss quantifies the difference between the predicted values and the true labels, with lower values indicating better performance. In the case of the Transformer, which employs a sequence-to-sequence model (encoder-decoder architecture), this loss can also be computed across the entire sequence of predicted words in translation tasks.

To minimize the error, the loss is propagated backward through the network using backpropagation. In this stage, the model calculates the gradient of the loss with respect to each parameter by applying the chain rule of calculus. These gradients are then used to update the model's weights. This controls the transformation of inputs to outputs. The Adam optimizer is a variant of stochastic gradient descent that is widely used. This is normally used for this purpose. Adam adjusts the rate of learning for each parameter, which helps increase the convergence speed and stability. The learning rate is changed dynamically during training using the following formula:

$$\mathrm{lr} = d_{\mathrm{model}}^{-0.5} \cdot \min\left(\mathrm{step\_num}^{-0.5}, \mathrm{step\_num} \cdot \mathrm{warmup\_steps}^{-1.5}\right) \tag{3.12}$$

Where d_{model} is the model's dimensionality and step num represents the current training step.
To prevent overfitting and improve the model's generalization ability, we have used regularization techniques, such as dropout and layer normalization. Dropout randomly turns off some of the neurons with each training step. This helps to prevent the model from becoming too dependent on a single neuron and can help to avoid overfitting. Layer normalization is applied to stabilize the learning process by normalizing the output of each layer and making the distribution of the activations stable across different layers and training steps. Pattern: The encoder and decoder layers are composed of multi-head self-attention and position-wise and feed-forward networks. The model can focus on different aspects of the input sequence at the same time to achieve this. This is especially applicable for long-range dependencies for translating complex sentences. The encoder produces a sequence of continuous representations of the input sequence. The decoder then uses it to create the corresponding output sequence one token at a time. The decoder self-attention mask makes it so that the model only looks at previous tokens and preserves the autoregressive nature of the sequence generation.

The effectiveness of the model is tested on a validation dataset, which is not part of the training set. The validation data set helps to evaluate the model on how well it generalizes to the unseen data. If the model works well on the training data, but works poorly on the validation data. This could be an indication of overfitting, which requires additional regularization or adjustments to the training process. Once the model has been trained, the final evaluation is done in a test dataset. The test dataset is not used during the training, is completely unseen, and is used to measure the generalization ability of the model. Various performance measures are calculated to determine how good the model is.

3.4.7 Key Performance Indicators for Sign to Text and Voice Generation

To evaluate our model, different performance metrics are quantitative measures used to evaluate the effectiveness and efficiency of a machine learning model. They provide a means to assess how well a model performs in terms of its ability to make accurate predictions on new, unseen data. Some commonly used performance metrics are given in the figure below:

i. Top-1 Accuracy
ii. Macro F1 Score
iii. Per-Class Precision, Recall, and F1 Score
iv. Confusion Matrix
v. Accuracy

Here's a detailed explanation of each performance metric used in the code:

i. Top-1 Accuracy: Top-1 accuracy refers to the percentage of times the most confident prediction (the top predicted class) made by the model matches the true label. This is the simplest and most commonly used metric for classification tasks. It answers: "What percentage of the time did the model's most confident prediction match the true class?". If the model predicts the correct class 85 times out of 100 predictions, the top-1 accuracy is 85%.

$$\text{Top-1 Accuracy} = \frac{\text{Number of correct predictions}}{\text{Total number of predictions}} \tag{3.13}$$

How it's calculated:

- The model predicts a class for each input sample.
- The predicted class is the one with the highest probability (confidence).
- If the predicted class matches the actual class, it is considered a correct prediction.
- Top-1 accuracy is the ratio of the number of correct predictions to the total number of predictions made.

ii. Macro F1 Score: The Macro F1 score is the average of the F1 scores for each class. It gives equal weight to each class. Regardless of the number of samples in each class. It is calculated by first computing the F1 score for each class and then averaging them. Precision answers: "How many of the predicted positive samples are actually positive?". Recall answers: "How many of the actual positive samples did the model successfully predict?".

(3.14)

$$\text{F1} = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}$$

The Macro F1 score is the average of the F1 scores for all classes. The macro F1 score is particularly useful when you have an imbalanced dataset. It treats each class equally, giving underrepresented classes the same weight as more common ones. It is a more balanced metric than accuracy in cases where the classes are imbalanced. It is calculated as:

$$\text{Macro F1} = \frac{1}{N}\sum_{i=1}^{N} \text{F1}_i \tag{3.15}$$

iii. Per-Class Precision, Recall, and F1 Score:

Precision measures the accuracy of positive predictions made by the model. It provides the answer to the question: “How many of the things the model says are positive are actually positive?” The accuracy is measured for each class. The positive predictive value is the proportion of a model that will predict positive correctly. This is a convenient approach when false positives need to be minimized. Recall is the ability of the model to find all relevant cases. It provides the answer to the question: "How many of the real positive items were the model able to identify?" Recall is calculated for each class, and it is a measure of the ability of the model to capture all the positive instances. It is a valuable tool when a low rate of false negatives is desired. F1 score is a harmonic mean of Precision and Recall. It provides an equal balance of the model’s accuracy, particularly when the data is imbalanced. The F1 score is a combination of both precision and recall. The F1 score gives equal weight to precision and recall. F1 is used when both false positives and false negatives are important. It is important to have a balance between the two. It is very helpful when dealing with imbalanced data where accuracy doesn't provide a good representation of the model’s performance.

iv. Confusion Matrix:

The confusion matrix is a table that is used to test the performance of a classification algorithm. It compares the actual class labels to the predicted class labels and aids in visualizing misclassifications. A confusion matrix for a 2-class classification problem looks like the given figure:

Table 3.9: Sample Confusion Matrix for Multiclass.

	Predicted: Ami	**Predicted: Pochondo**	**Predicted: Baba**
True: Ami	TP	FN	FP
True: Pochondo	FP	TP	FN
True: Baba	FN	FP	TP

In a multi-class problem, the matrix becomes square. The matrix, each row is the actual class, and each column is the predicted class. It gives an indication of the performance of the model in classifying the instances of each class. Misclassifications are off-diagonal and can be used to compute the values of precision, recall, accuracy, and F1 score.

V. Accuracy

Accuracy is a commonly used performance metric in machine learning and deep learning. It measures the proportion of correct predictions made by a model out of all the predictions made. In other words, accuracy tells us how often a model is correct when making a prediction. Accuracy is calculated by dividing the number of correct predictions made by the model by the

total number of predictions made. It is expressed as a percentage ranging from 0 to 100, where 100 represents perfect accuracy (i.e., all predictions made by the model are correct).

Table 3.10: Performance Metrics and Their Equations.

Metric	**Equation**
Accuracy	$\text{Accuracy} = \frac{\text{Number of correct predictions}}{\text{Total number of predictions}}$
Precision	$\text{Precision}_i = \frac{\text{True Positives}_i}{\text{True Positives}_i + \text{False Positives}_i}$
Recall	$\text{Recall}_i = \frac{\text{True Positives}_i}{\text{True Positives}_i + \text{False Negatives}_i}$
F1 Score	$\text{F1}_i = 2 \cdot \frac{\text{Precision}_i \cdot \text{Recall}_i}{\text{Precision}_i + \text{Recall}_i}$
Macro F1 Score	$\text{Macro F1} = \frac{1}{N} \sum_{i=1}^{N} \text{F1}_i$
Confusion Matrix	A table summarizing TP, FP, TN and FN for each class.
Top-1 Accuracy	$\text{Top-1 Accuracy} = \frac{\text{Number of correct top predictions}}{\text{Total number of predictions}}$

3.5 SUMMARY

In this study, a real-time Bangla Sign Language (BSL) recognition and 3D avatar generation system was developed to facilitate bidirectional communication between hearing and hearing-impaired individuals. The system consists of two main components: converting Bangla speech into animated sign language and recognizing BSL (Bangla Sign Language) gestures to convert them into text or voice. The first component captures spoken Bangla using the Google Speech API. It converts speech to text. This text is then tokenized, parsed, and lemmatized. This lemmatized text is then used to identify the corresponding BSL sign. So if a sign is not found in the database, the system breaks it down to the individual letters. Then it creates respective animations using a 3D avatar. The second component is used to process sign language gestures from videos. It uses deep learning models to extract keypoints using MediaPipe and then recognize the gestures. The training dataset contains a variety of signers, different environments, and multiple devices. This diversity ensures robust performance thanks to this diversity. The system looks at accuracy for the two components, voice to sign and sign to voice, using performance measures of accuracy, precision, and recall, and using expert ratings for sign language animations. By combining state-of-the-art linguistic processing, deep learning, and 3D animation, our system offers an efficient and scalable solution for enhancing accessibility. It offers real-time communication and facilitates the gap between the hearing and hearing-impaired communities in Bangladesh.

CHAPTER 4

RESULTS AND DISCUSSION

4.1 INTRODUCTION

The chapter presents the results of our research work on a real-time BSL recognition and translation system. Our research work brings about two-way communication between the hearing and the deaf. The system has two basic elements: voice-to-sign translation and sign-to-text/voice conversion. In this chapter, the results of the voice-to-sign language are presented first. After that, the results of the simulation on sign-to-voice and text translation are presented.

4.2 VOICE TO SIGN CONVERSION

In this section, we will discuss the evaluation outcomes of the voice-to-sign animation component of the proposed system. The results focus on four categories. These categories are letters, numbers, words, and sentences. They together represent the core elements of daily communication in Bangla Sign Language. Each category was assessed by seven professional sign language experts. We have used a five-point Likert scale ranging from Poor to Excellent. The evaluation aimed to measure the accuracy, clarity, fluency, and naturalness of the generated 3D avatar animations. In this result analysis, we have identified both the strengths of the system in producing accurate and meaningful signs. We also ensure natural transitions and sentence-level fluency, and the time that has taken to show a sign.

4.2.1 Alphabets Evaluation

Alphabets constitute the foundation of sign language communication. It is often used for spelling names, technical terms, or words without direct signs. Therefore, ensuring accuracy in their animation is necessary for effective BSL interpretation. In this study, the 26 English alphabets (A-Z) were evaluated by seven expert respondents using a five-point Likert scale (1 = Poor, 5 = Excellent). The evaluation aimed to measure clarity, naturalness, and accuracy of animations through respondent-wise and alphabet-wise analyses. The evaluation achieved an average score of 4.63 out of 5, corresponding to an average accuracy of 92.53%. This indicates that the majority of alphabet animations were perceived as accurate and effective for communication.

As shown in Figure 4.1: Respondent-wise average accuracy for alphabet animations, the ratings remained consistently high, with average scores ranging from 4.2 to nearly 5.0. This reflects strong consensus among experts on the overall quality of the alphabet animations

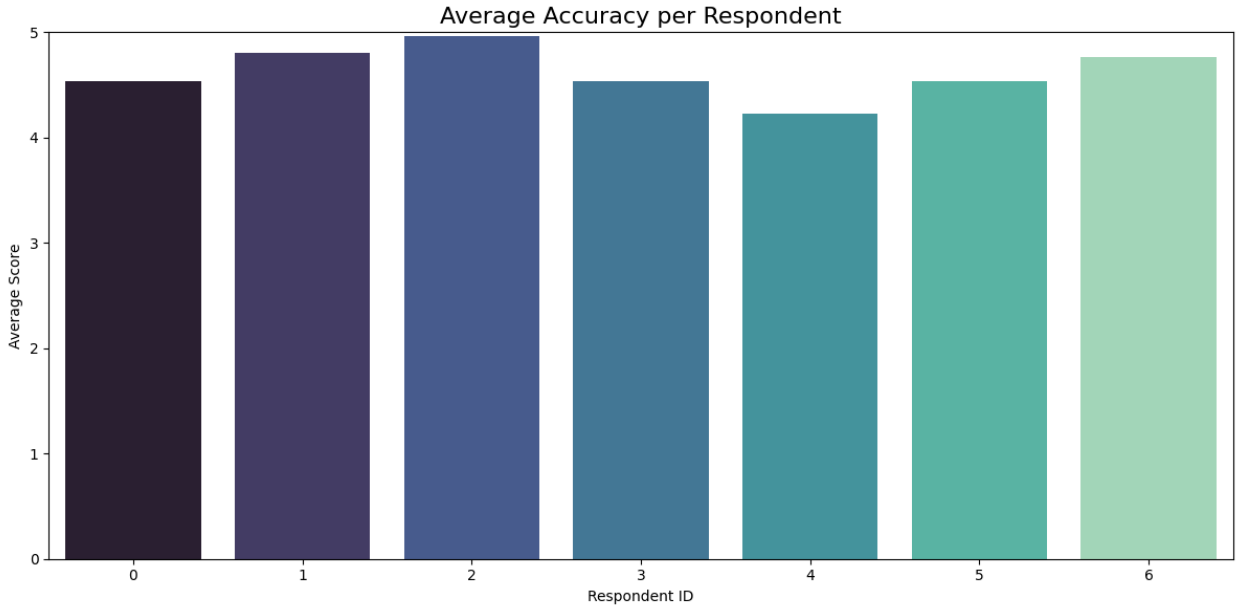


Figure 4.1: Respondent-wise average accuracy for alphabet animations.

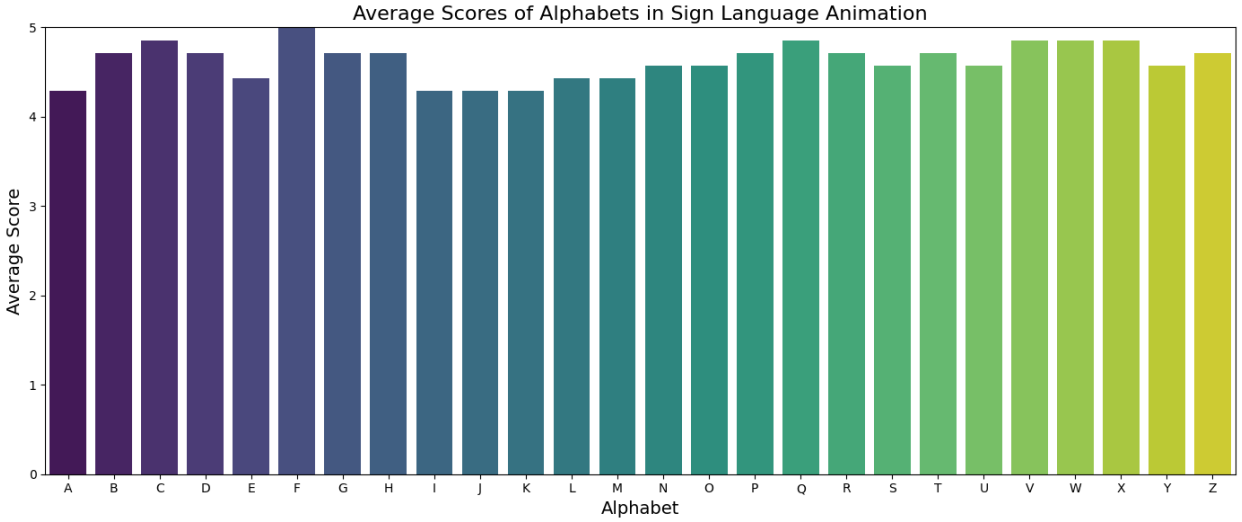


Figure 4.2: Average scores of individual alphabets (A–Z) in Bangla Sign Language animation.

Figure 4.2 shows that several alphabets, such as *C, F, W, X,* and *Y*, achieved near-perfect scores, while a few, including *A, E, I, J,* and *K*, scored slightly lower (around 4.2–4.3). These differences suggest that although most gestures were highly accurate, certain signs may require refinement for improved clarity.

The ranking of the alphabet by difficulty is illustrated in 4.3, where *A, E, I, J,* and *K* were identified as the most challenging to interpret, while *F, W, V,* and *Q* were rated the most accurate and natural. That means there is no need for correction for these letters. They are doing up to the mark.

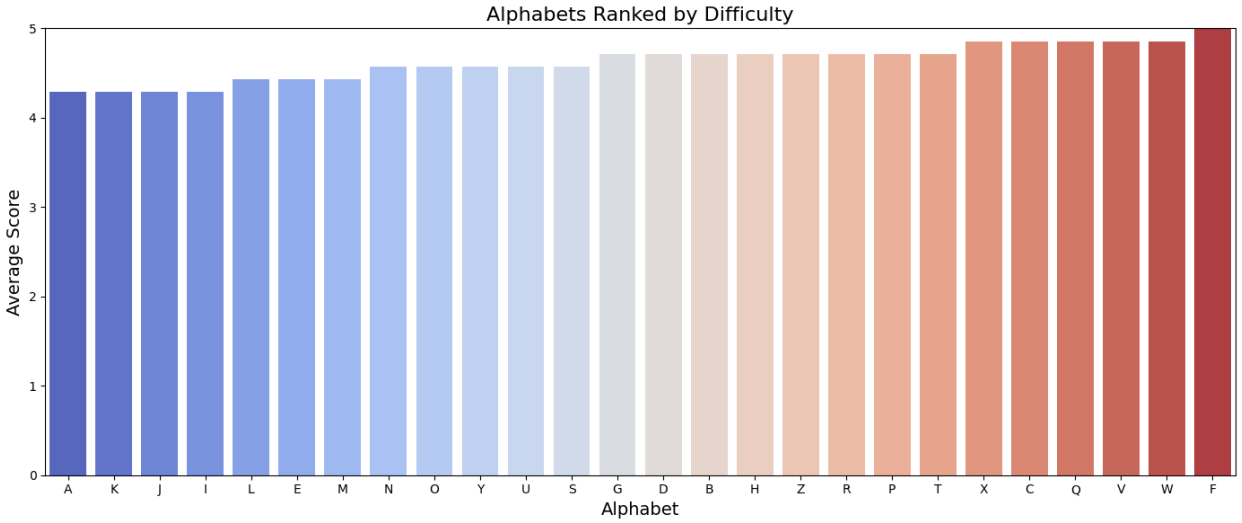


Figure 4.3: Difficulty ranking of alphabets based on expert evaluation.

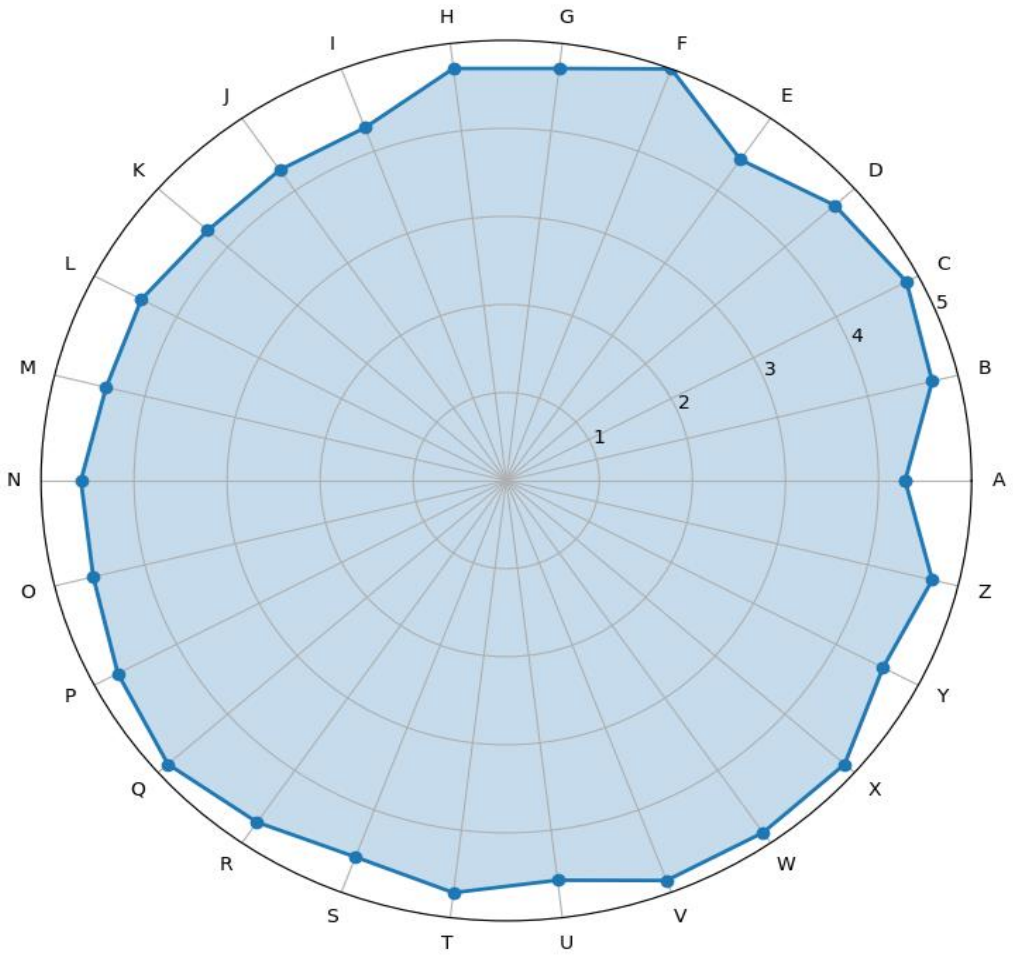


Figure 4.4: Radar chart of average scores for alphabet animations.

The above figure demonstrates the uniformity of performance, with most letters forming a near-complete circular pattern close to the maximum score. This indicates strong consistency across the dataset, with only minor deviations for a few letters. The evaluation confirms that alphabet animations achieved

92.53% accuracy overall, with strong consensus among respondents. While most letters were rated highly, a small subset (*A, E, I, J,* and *K*) may benefit from further refinement.

4.2.2 Numbers Evaluation

In this study, the developed Bangla Sign Language (BSL) number animations for digits 0–9 were evaluated by seven expert respondents using a five-point Likert scale (1 = Poor, 5 = Excellent). The evaluation aimed to assess the clarity, naturalness, and overall effectiveness of the animations. The results are presented through respondent-wise, digit-wise, and difficulty-based analyses, supported by statistical summaries and visualizations. This evaluation not only validates the effectiveness of the system but also identifies specific areas where refinements may be required to enhance gesture accuracy.

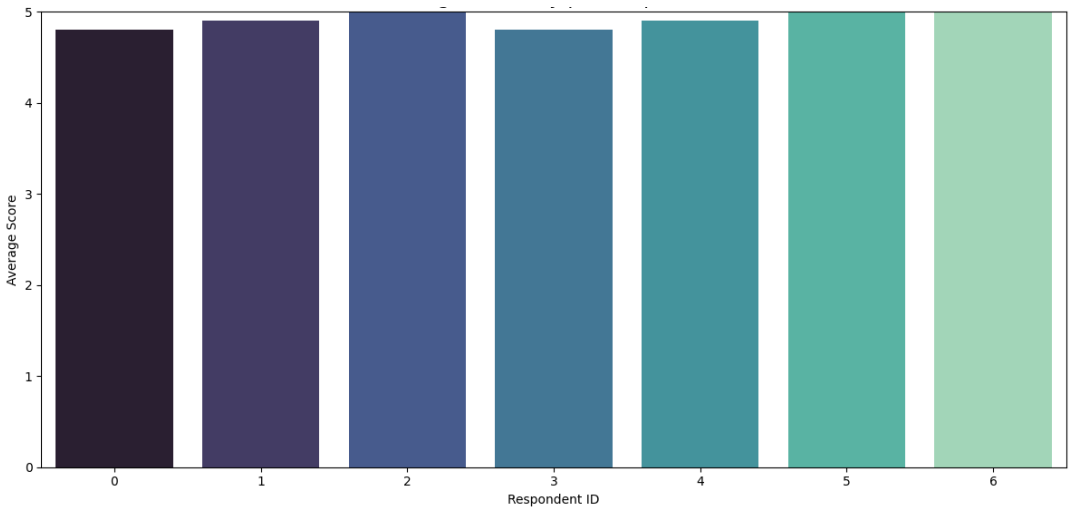


Figure 4.5: Respondent-wise average accuracy for number animations.

The above Figure 4.5 illustrates the average accuracy achieved by each of the seven expert respondents. All respondents consistently rated the animations between 4.8 and 5.0 on average, demonstrating a strong agreement and minimal variation in their evaluations. The figure highlights the reliability of the developed system across different evaluators.

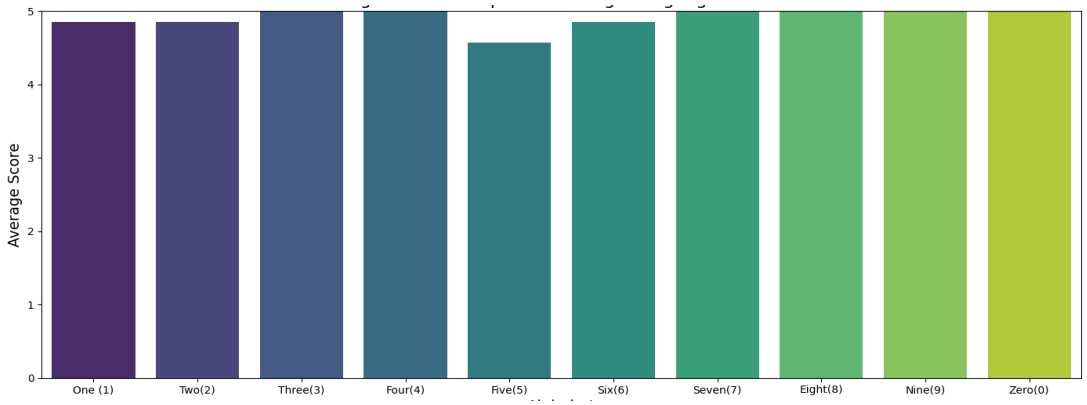


Figure 4.6: Digit-wise average scores for Bangla Sign Language number animations (0–9).

The above figure presents the average Likert-scale scores (1–5) assigned to each digit by the expert respondents. Most digits, including 0, 7, 8, and 9, received nearly perfect ratings, reflecting their clarity

and naturalness. Slightly lower scores (~4.6) were observed for 5, indicating that this digit may require further refinement for improved recognition.

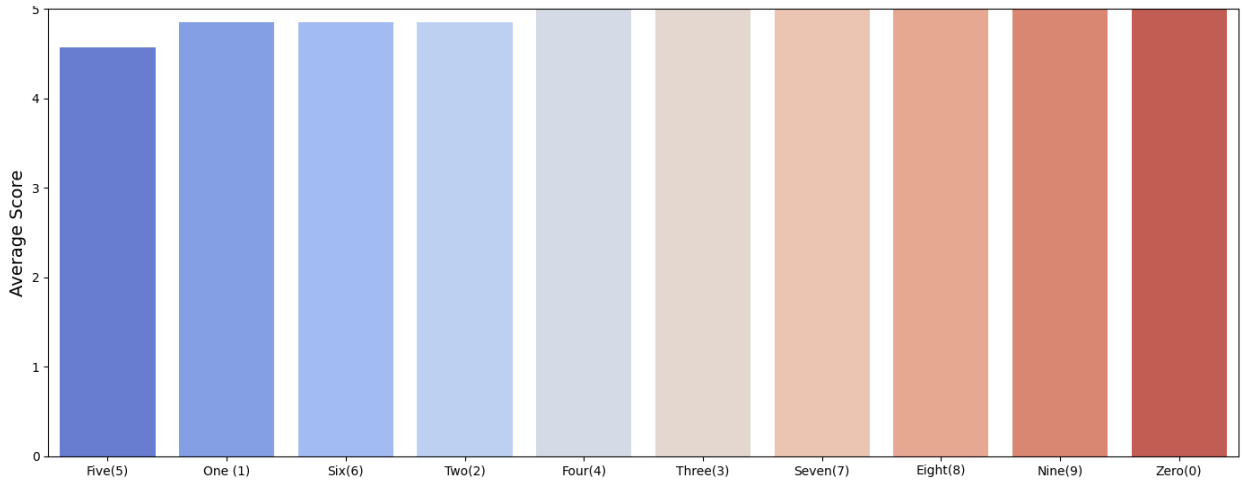


Figure 4.7: Difficulty ranking of digits in number animation evaluation.

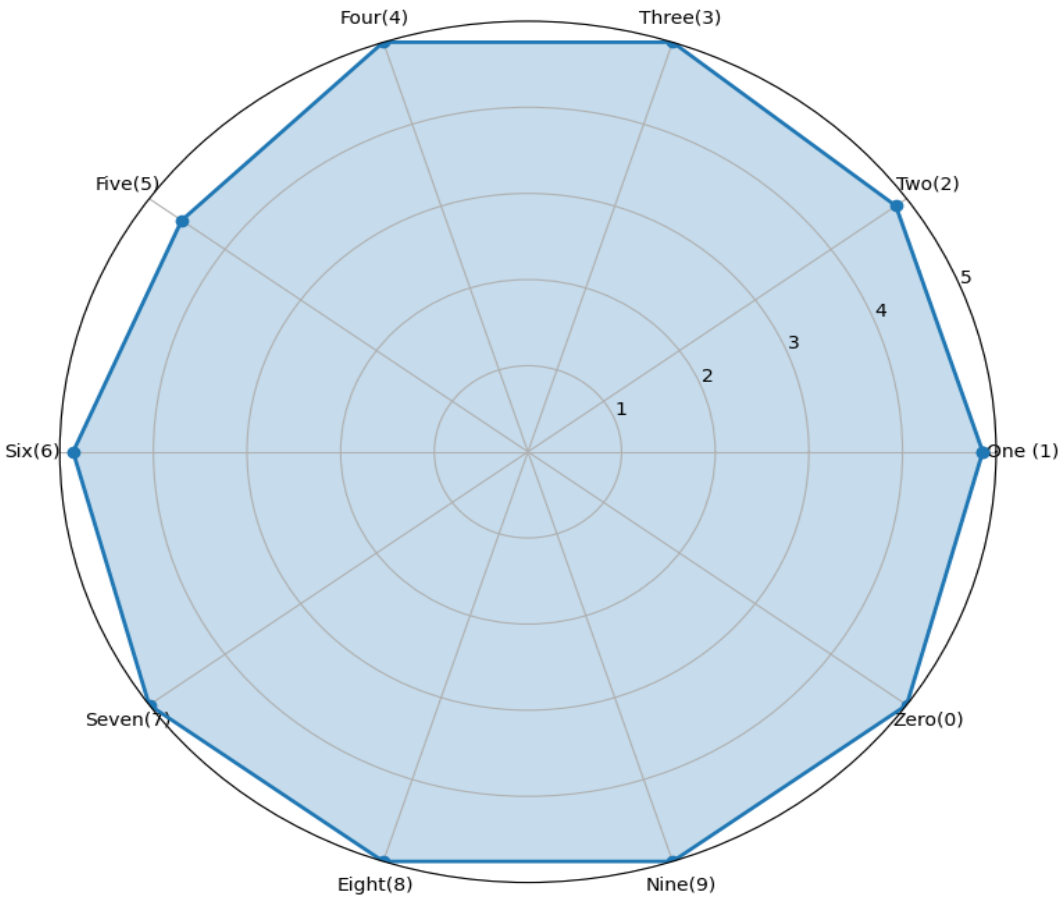


Figure 4.8: Radar chart of average scores for digits (0–9).

This Figure 4.7 displays the ranking of digits based on their perceived difficulty, as judged by the experts. Five (5) was rated as the most challenging to interpret accurately, followed by One (1) and Six (6). In contrast, digits such as 0, 7, 8, and 9 were found to be the easiest, achieving the highest scores

with minimal ambiguity.

This radar Figure 4.8 provides a holistic view of the performance across all digits. The near-complete circular shape, with most values clustered close to the maximum rating, indicates uniformity and consistency in the animations. Small deviations for some digits (e.g., 5) are visible, reaffirming the need for targeted refinements. The average score for the numbers is 4.91 out of 5, and the average accuracy is 98.29%.

4.2.3 Word Evaluation

Our developed Bangla Sign Language (BSL) word animations were evaluated by seven expert respondents. We have used a five-point Likert scale (1 = Poor, 5 = Excellent). The evaluation aimed to assess the clarity, naturalness, and overall effectiveness of the animations. The results are described through respondent-wise, word-wise, and difficulty-based analyses. This is done by statistical summaries and visualizations. This evaluation not only validates the effectiveness of the system but also identifies specific areas where refinements may be required to enhance gesture accuracy.

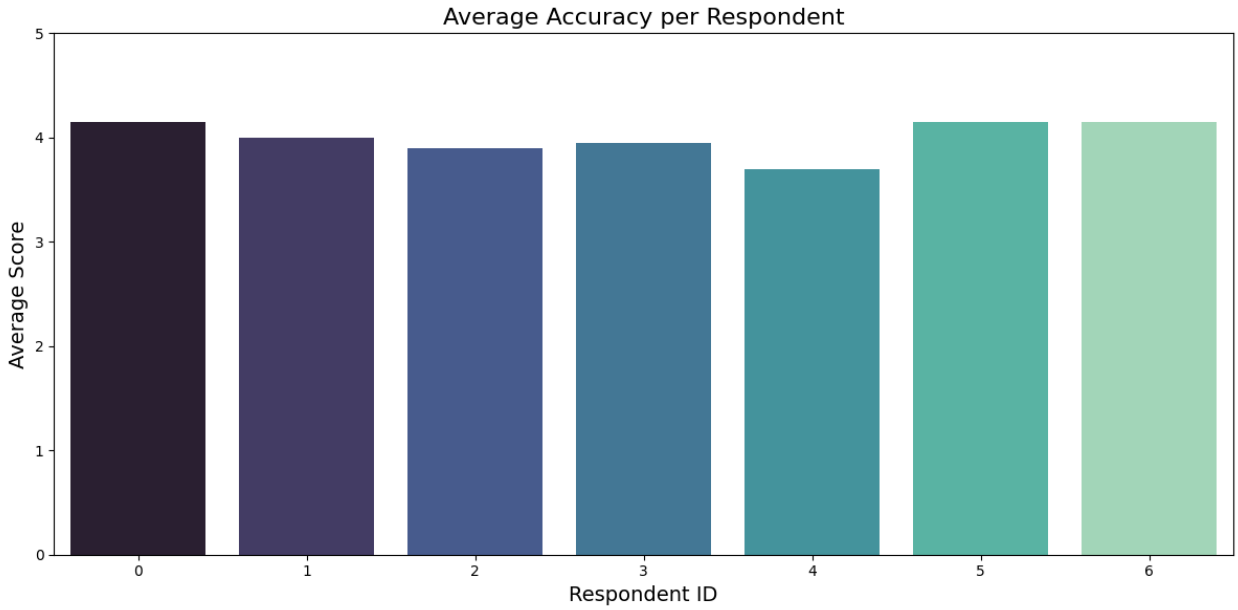


Figure 4.9: Respondent-wise Average Accuracy for Word Animations.

The above bar plot shows the average accuracy achieved by each of the seven expert respondents. In this plot, the scores range from 3 to 5, showing a strong agreement on the quality of most sentences. Here are a few sentences that received slightly lower scores. This indicates that refinements are needed for clearer interpretation.

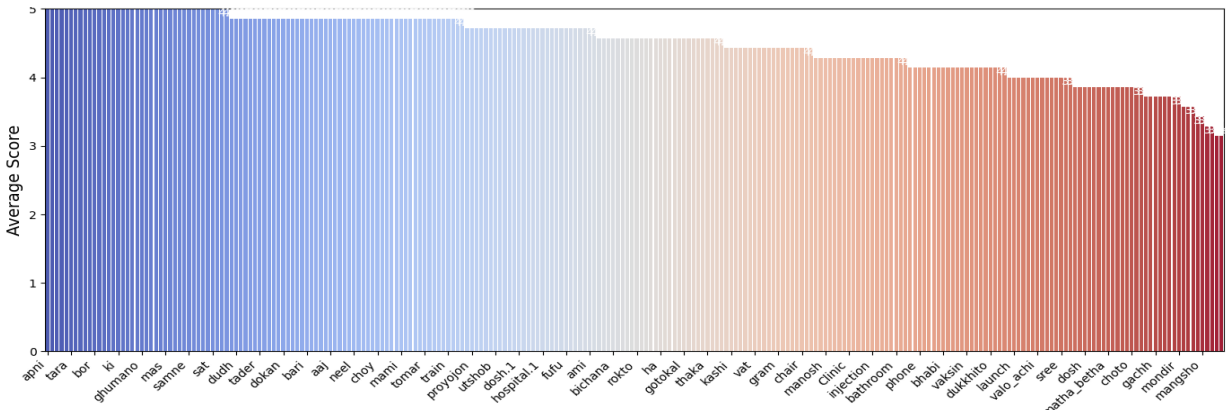


Figure 4.10: Word-wise Average Scores for Bangla Sign Language Word Animations.

The above figure 4.10 represents the average Likert-scale scores from 1 to 5. This was assigned to each sentence by the expert respondents. Words such as "apni" and "tara" received near-perfect ratings. That means these words reflect their clarity and naturalness. But the sentences like "mondir" and "mangsho" had slightly lower average scores (around 3.7). This indicates a space for improvement in their recognition.

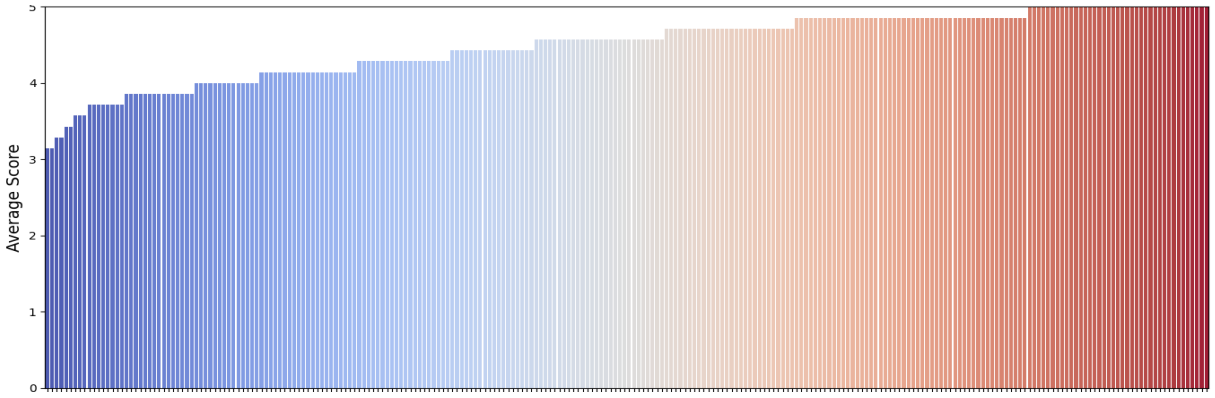


Figure 4.11: Difficulty Ranking of Words in Animation Evaluation.

The above bar plot, Figure 4.11, displays the ranking of sentences based on their perceived difficulty. This was judged by the 7 experts. Sentences like "owsod" and "tumi" were found to be the easiest. They receive higher scores with minimal ambiguity. On the other hand sentences such as "debor" and "jilapi" were considered more challenging, as they exhibited a broader range of interpretations.

The above radar Figure 4.12 provides a holistic view of the performance across all sentences. Most values are clustered close to the maximum rating. This indicas consistency in the animations. Some deviations for words such as "mach" reflect the areas that require targeted refinements to ensure clarity. The evaluation of the 250 word Bangla Sign Language (BSL) animations confirms an overall accuracy of 82.94%. This indicates a strong performance across the dataset. The words received a mean score of 4.1469 on a 5-point Likert scale. This result suggests that the majority of the signs were rated positively by the respondents.

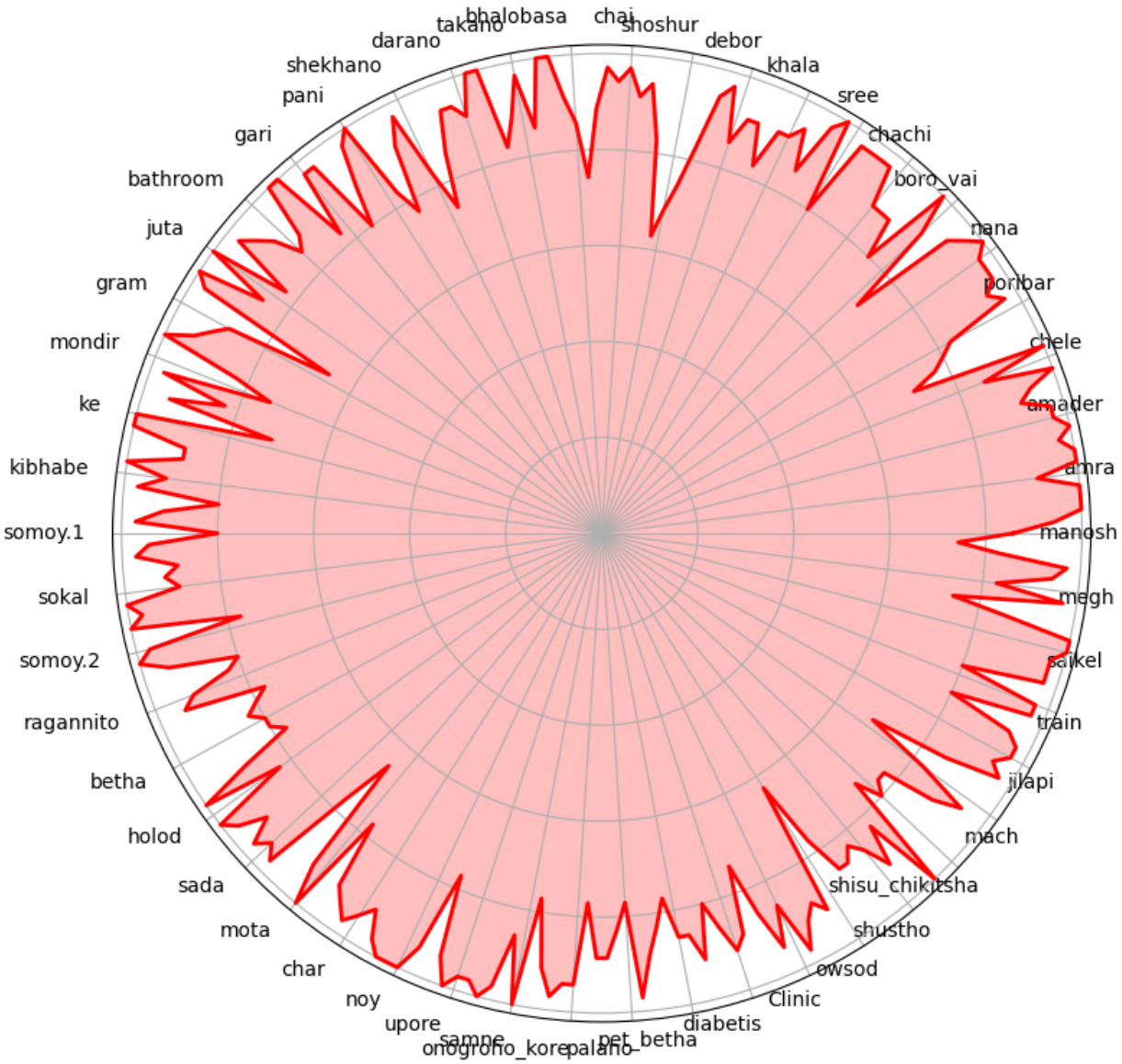


Figure 4.12: Radar Chart of Average Scores for Sentences.

4.2.4 Sentence Evaluation

Our developed Bangla Sign Language (BSL) sentence animations were evaluated by seven expert respondents using a five-point Likert scale (1 = Poor, 5 = Excellent). The evaluation aimed to assess the clarity, naturalness, and overall effectiveness of the animations. The results are described through respondent wise, sentence wise, and difficulty-based analyses. This is supported by statistical summaries and visualizations. This evaluation validates the effectiveness of the system and identifies specific areas where refinements may be required to enhance gesture accuracy.

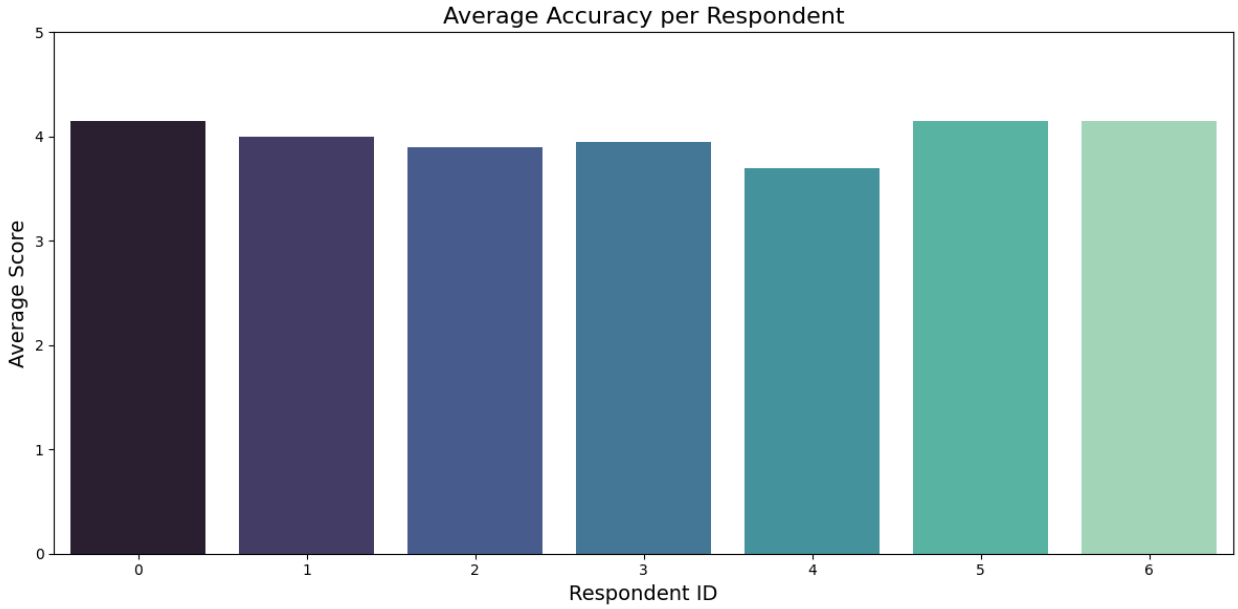


Figure 4.13: Respondent-wise average accuracy for sentence animations.

The above bar plot shows the average accuracy achieved by each of the seven expert respondents. The scores range from 3 to 5, showing a strong agreement on the quality of most sentences. However, a few sentences received slightly lower scores, indicating that refinement may be needed for clearer interpretation.

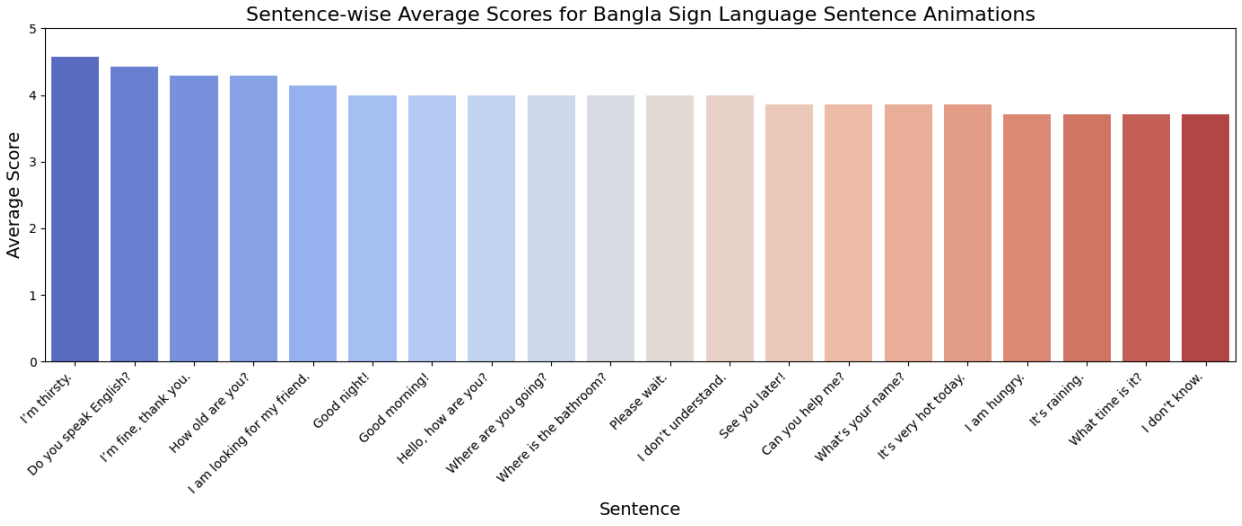


Figure 4.14: Sentence-wise average scores for Bangla Sign Language sentence animations.

This figure presents the average Likert-scale scores (1–5) assigned to each sentence by the expert respondents. Sentences such as "Good morning!" and "Where are you going?" received near-perfect ratings, reflecting their clarity and naturalness. On the other hand, sentences like "I am hungry." and "It's raining." had slightly lower average scores (around 3.7), indicating room for improvement in their recognition.

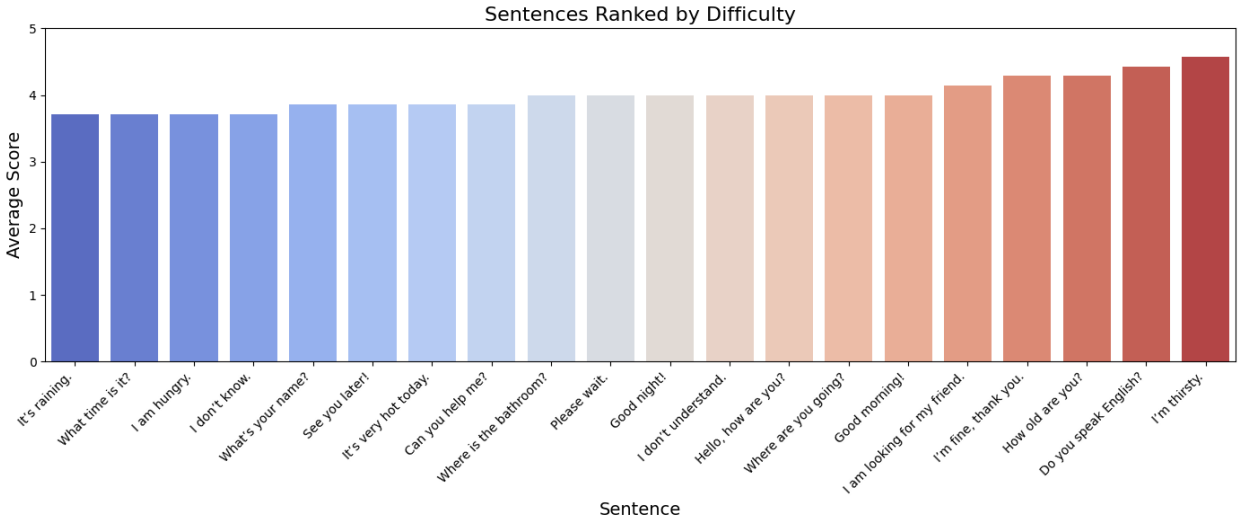


Figure 4.15: Difficulty ranking of sentences in animation evaluation.

This bar plot displays the ranking of sentences based on their perceived difficulty, as judged by the experts. Sentences like "What time is it?" and "Where is the bathroom?" were found to be the easiest, receiving higher scores with minimal ambiguity. In contrast, sentences such as "I don't understand" and "I am looking for my friend" were considered more challenging, as they exhibited a broader range of interpretations.

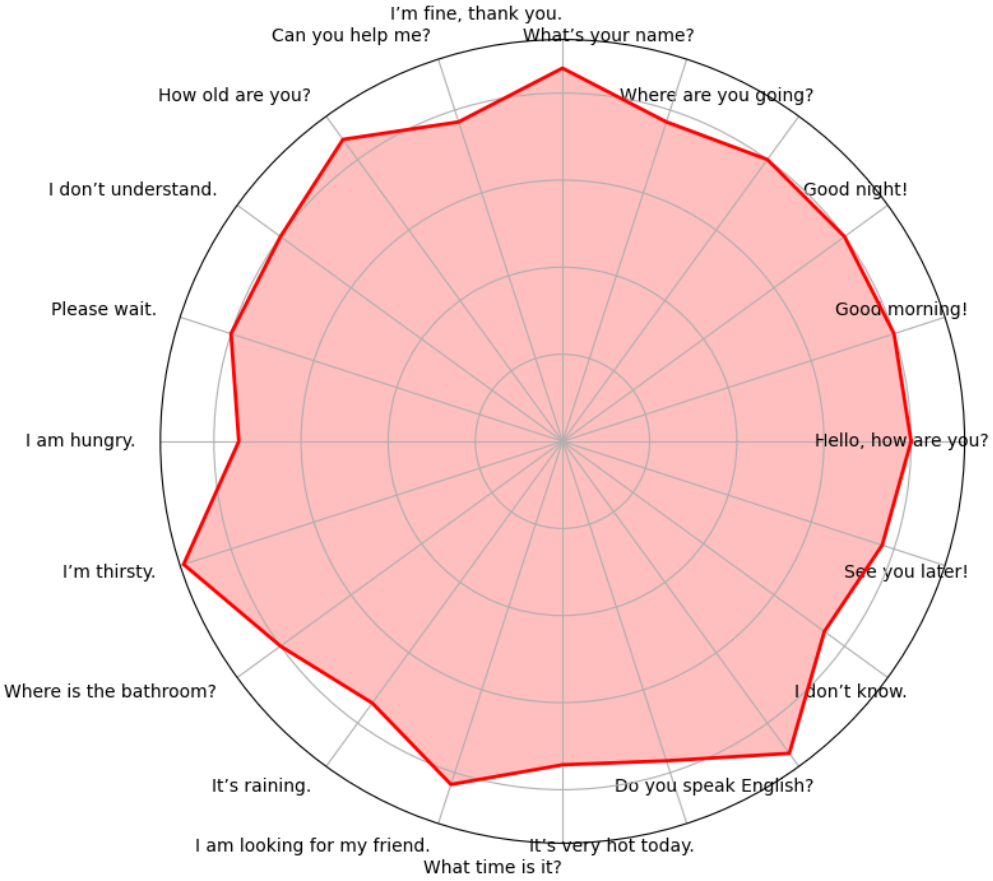


Figure 4.16: Radar plot of average scores for our created sentences.

The above radar plot provides a holistic view of the performance across all sentences. Most values are clustered close to the maximum rating, indicating consistency in the animations. For sentence animations, the system's performance was again strong, with a mean score of 4.11 and the accuracy is 82.20%. Some minor deviations for sentences such as "I don't know." reflect the areas that require targeted refinements to ensure clarity.

4.2.5 CPU Time Analysis for Sign Language Generation System

The CPU time for processing each word in the Bangla Sign Language recognition system was analyzed to understand the computational complexity of recognizing various signs. This analysis is essential for optimizing performance and identifying potential bottlenecks in the recognition pipeline. The dataset consists of various sign language words and the corresponding CPU time. The CPU time represents the amount of time the system takes to recognize a particular sign.

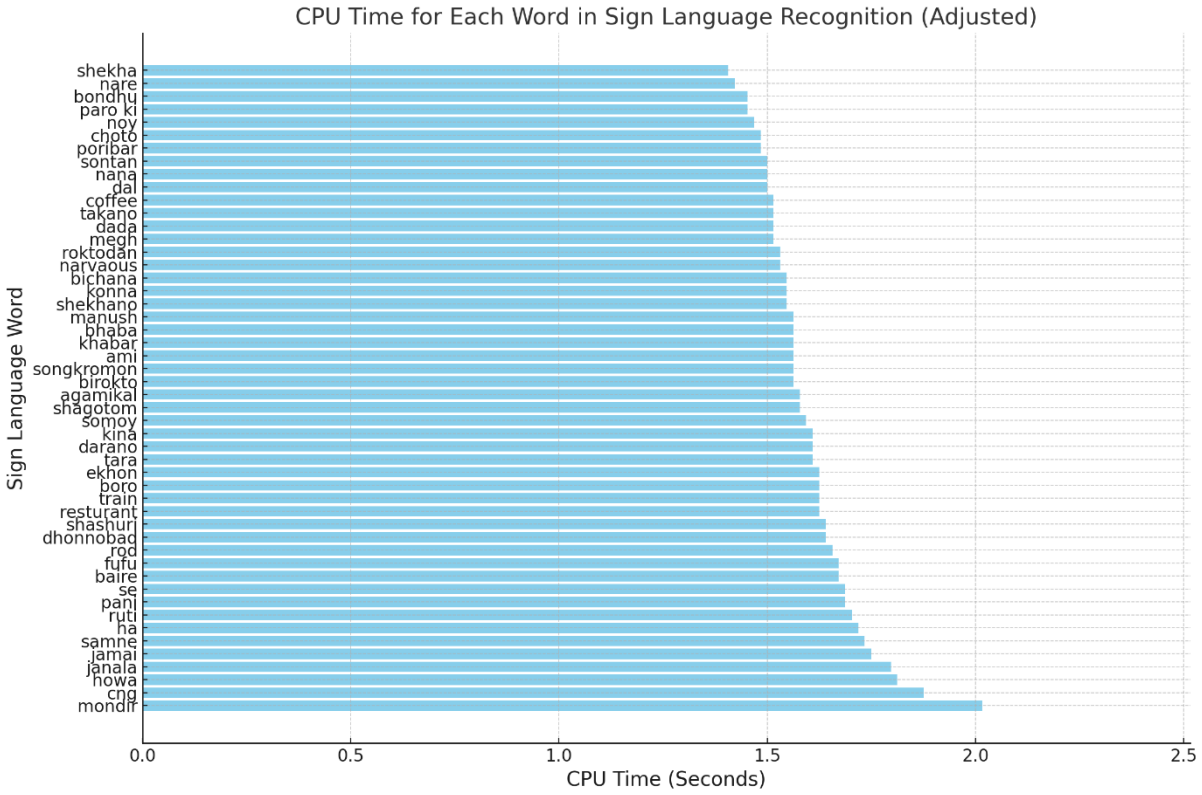


Figure 4.17: CPU time for individual words.

Figure 4.16 reveals considerable variation in the CPU time required for different words. Some words require significantly more processing time than others. Due to various factors such as the complexity of the sign, frames processed, or some similar gestures. Some words such as "janala" and "mondir", take notably longer to process. These words might involve more intricate hand movements or are harder for the system to differentiate based on the dataset or model used. Conversely, words like "bondhu", "nare", and "noy" are processed relatively faster. Due to simpler movements or more distinct features in the dataset, the system can easily recognize. The bar plot above visualizes the CPU time required to process each word. The x-axis represents the CPU time in seconds, and the y-axis lists the words. To make the figure understandable, we didn't show all the words available in our database. Words are sorted by their CPU time, with the more complex words appearing at the top of the graph. By skipping certain labels, the figure provides a clearer view of the processing times. This allows for a better comparison of the

words with the longest and shortest processing times. The average processing time is 1.58 seconds.

4.2.6 Summary of Voice-to-Sign Evaluation

This section presents the evaluation of the BSL voice-to-sign animation system. This covers alphabet, numbers, words, and sentences for everyday communication. The system's effectiveness was assessed by seven professional sign language experts using a five-point evaluation process. The key focus was on the clarity, fluency, accuracy, and naturalness of the 3D avatar animations. The evaluation of 26 English alphabets (A-Z) revealed an average score of 4.63/5, translating to an accuracy of 92.53%. Some letters like "C", "F", "W", "X", and "Y" achieved near-perfect scores. The alphabet "A", "E", "I", "J", and "K" were rated slightly lower (around 4.2-4.3). This indicates minor areas for improvement. BSL number animations for digits 0-9 also performed well, with average scores between 4.8 and 5.0 across all experts. The digit "5" was the most challenging, receiving a lower score (~4.6). A radar chart confirmed the consistency of the animations, though some digits, like "5", need refinement. The evaluation highlighted that digits 0, 7, 8, and 9 were the easiest to interpret. The evaluation of 250 BSL words led to a mean score of 4.1469, with an overall accuracy of 82.94%. While most words received high scores, words like "mondir" and "mangsho" had slightly lower scores (~3.7). For sentence animations, the system's performance was again strong, with a mean score of 4.11, and the accuracy is 82.20%. Sentences like "Good morning!" and "Where are you going?" were highly rated, while sentences such as "I am hungry" and "It's raining" were rated lower (~3.7). The average CPU time was 1.58 second.

Table 4.1: Summary table for different signs accuracy and CPU time.

Category	Average Score	Accuracy
Alphabets	4.63 (Out of 5)	92.53%
Numbers	4.91 (Out of 5)	98.29%
Words	4.15 (out of 5)	82.94%
Sentences	4.11 (out of 5)	82.20%
CPU Time	1.58 Seconds	

4.3 SIGN TO VOICE AND TEXT CONVERSION

This section presents the results and a comprehensive discussion of the performance. We have discussed the Sign to Voice and Text Conversion system developed for recognizing BSL animations. The goal of this system is to translate BSL gestures into both text and voice outputs. This makes sign language more accessible and enables seamless communication between sign language users and non-sign language users. The evaluation of the model is based on a set of performance metrics, including accuracy, F1-score, and confusion matrix. The performance metrics are the indicators of how well a model is performing. This measures the effectiveness of the system in recognizing and interpreting the signs. This section also emphasizes the key challenges. This has problems concerning data imbalance, sign visual ambiguity, and misclassifications. The analysis of these challenges provides an insight into areas for improvement. We have improved some areas. We have taken advantage of more signer diversity and

improved preprocessing techniques. The effectiveness of the sign-to-text and sign-to-voice conversions of the system will be discussed in the following subsections. It provides insights about the limitations of the system and gives you an idea about how to make the model more efficient in future iterations. The results aim to guide other developments in the field of sign language recognition and to integrate it into real-world applications. This research work will increase the accessibility of communications for the Deaf community.

4.3.1 Model Performance on Validation and Test Data

In evaluating the effectiveness of the developed Sign to Voice and Text Conversion system. The performance of the model was measured mainly on two important factors. They are the validation accuracy and the Macro F1-score. These metrics can provide valuable insights into how well the model is able to generalize to unseen data. It also balances its performance across different classes. This is very important in sign language recognition. The overall validation accuracy of this model was 83.24% on the validation set.

i. Validation Accuracy: This value is the percentage of correct predictions on the validation data out of the total number of predictions made on the validation data. The relatively high accuracy score shows that the model has been able to learn generalizations on unseen data. It is also capable of accurately recognizing a wide variety of sign language gestures. To evaluate the model's performance in a better manner. It is important to compare this accuracy with the baseline performance or the performance of other models that are trained on similar datasets. The validation accuracy of 83.24% indicates that the model is performing well as compared to the previous studies in the domain of sign language recognition. There is still room for improvement. The classes with lower sample sizes or with visual ambiguities.

ii. Macro F1-Score

The Macro F1-score is another important metric that is used to evaluate the performance of the model. The Macro F1-score is the arithmetic mean of the F1-scores for all classes. F1-score is the harmonic mean of precision and recall, and it is a compromise between the ability of the model to identify relevant classes and its ability to identify all relevant instances. In the case of the validation set, the model achieved a Macro F1-score of 0.7806, suggesting a relatively good balance in recognizing different classes. We find that on average, the model performs well across all classes and does not show any extreme biases toward any particular class. But the model's performance varied between classes, with some gestures being recognized with higher accuracy and others with lower F1-scores.

Table 4.2: Top-10 Best Performing and Worst Performing Classes.

	Class	Label	Precision	Recall	F1
TOP 10	0	Aaj	1.0	1.0	1.0
	38	Debor	1.0	1.0	1.0
	32	Coffee	1.0	1.0	1.0
	27	Chele	1.0	1.0	1.0

CLASS	33	Cycle	1.0	1.0	1.0
	24	Chachi	1.0	1.0	1.0
	19	Boro Vai	1.0	1.0	1.0
	35	Dadi	1.0	1.0	1.0
	36	Dal	1.0	1.0	1.0
	14	Bon	1.0	1.0	1.0
WORST 10 CLASS	70	Kolom	0.0	0.0	0.0
	44	Dorja	0.0	0.0	0.0
	46	Dudh	0.0	0.0	0.0
	60	Hatha	0.0	0.0	0.0
	57	Gorom_food	0.0	0.0	0.0
	2	Agamikal	0.0	0.0	0.0
	122	Taka	0.0	0.0	0.0
	84	Nonod	0.0	0.0	0.0
	120	Table	0.0	0.0	0.0
	119	Sree	0.0	0.0	0.0

The above table displaying the top-10 best-performing classes and the worst-10 performing classes, including their accuracy scores, would provide clarity on how well the model performs across different signs.

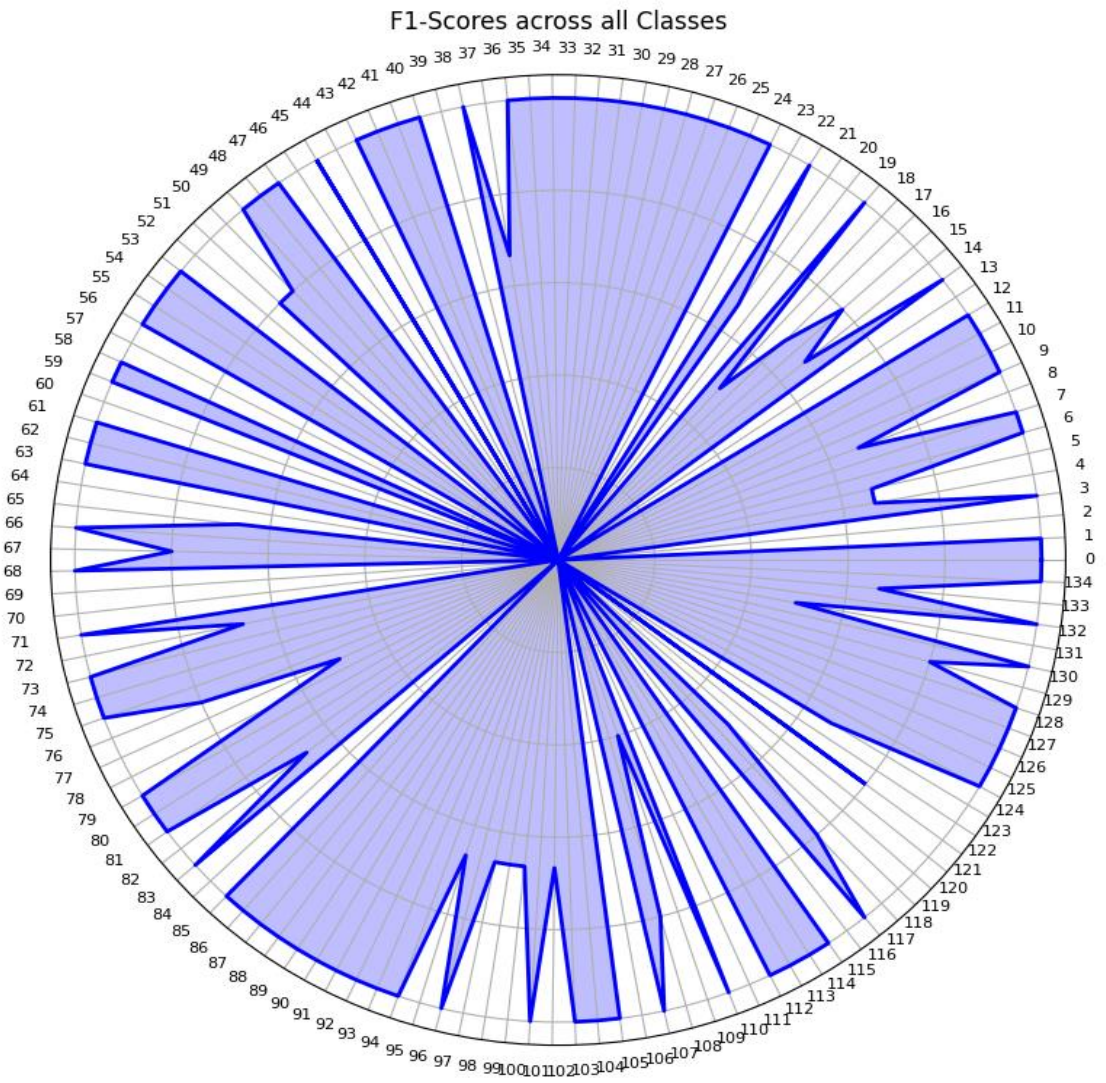


Figure 4.18: Spider Plot for the F1-scores of all classes.

A radar chart or spider plot showing the F1-scores of the individual classes can visually represent how well the model performs for each sign language gesture. The chart can help highlight areas where the model struggles and which classes have a higher F1-score. From the figure, it's evident that some classes performed exceptionally well with perfect scores. For instance "Aaj", "Debor", and "Coffee" all achieved a perfect Precision, Recall, and F1-score of 1.0.

However, there were a few worst classes like "Taka", "Shalika", "Dekha", "Ful", and "Sree", which recorded F1-scores of 0.0. The Precision and Recall for these classes were also 0.0, which means that the model was not able to successfully identify any samples from these classes, indicating that there may be some issues with the data being imbalanced or with the data not being well represented in the training set.

4.3.2 Training Curves

This section presents an in-depth analysis of the model's performance during the training process. To be specific, it focuses on the loss, accuracy, and Macro F1-score metrics across 100 epochs. These metrics were tracked for both the training and validation datasets to monitor the model's learning

progression and generalization capability.

i. Training Loss:

The training loss is definitely decreasing monotonously, from 5.0140 at epoch 1 to 0.4583 at epoch 100. This continual loss reduction shows that the model was learning and optimizing with time. A decreasing loss value reflects the model's improved ability to minimize the error between its predictions and the true labels, suggesting that the training process was successful and stable. From the figure below, we can see that in epoch 1, the loss was 5.0140. This indicates a high initial error rate. Epoch by epoch the loss decreases. In epoch 100, the loss becomes 0.4583. This reflects a significant reduction in error as training progressed. The steady decline with no major fluctuations suggests that the learning rate and optimization strategies were effective in reducing error without causing instability.

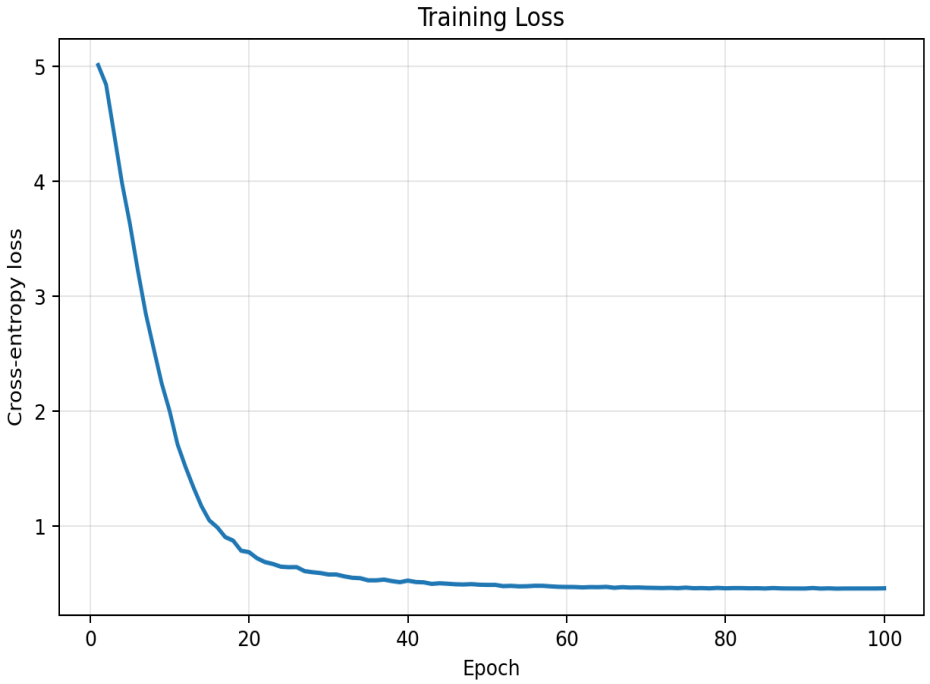


Figure 4.19: Training Loss Curve.

ii. Training Accuracy:

The training accuracy also improved steadily over the epochs. Starting from a low value of 0.008 at epoch 1, it rose to 0.997 by the final epoch (epoch 100). This rapid increase in training accuracy shows that the model quickly adapted to the data and became highly proficient at recognizing patterns in the training set.

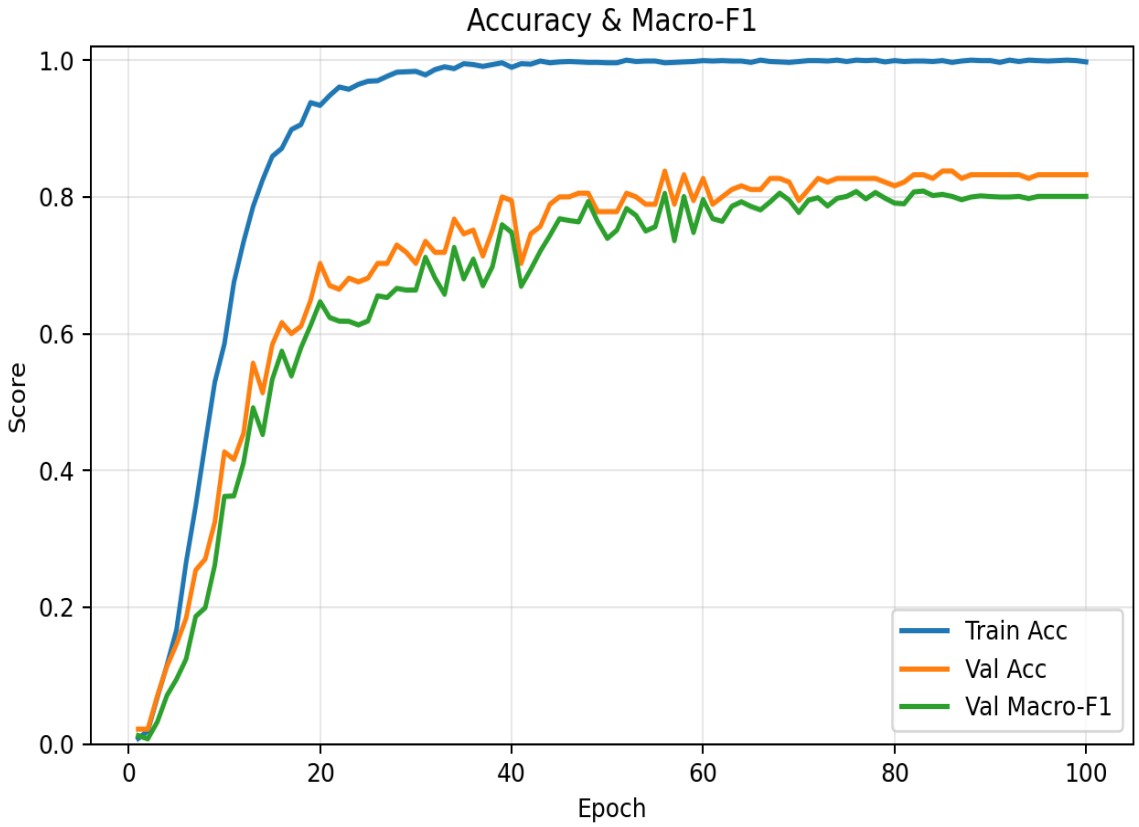


Figure 4.20: Taring-Validation-Validation Macro-F1 Accuracy.

In 1st epoch Training Accuracy was 0.008%. The accuracy increases epoch by epoch. In the 100th epoch the Training Accuracy becomes 99.7%. The fast improvement in training accuracy, particularly after the first few epochs, reflects the model's learning process, where it began to correctly classify samples more frequently as it encountered more examples.

iii. Validation Accuracy

The validation accuracy provides insight into how well the model generalizes to unseen data. Starting at 0.022 in the first epoch, the validation accuracy steadily increased to 0.832 by epoch 100. This gradual increase demonstrates that the model was able to generalize well, achieving high performance on the validation set by the end of the training. Notably, there were no major discrepancies between the training and validation accuracy curves, which indicates that the model did not overfit. The model successfully applied the patterns it learned from the training data to the unseen validation set, as reflected in the steady improvement across epochs.

iv. Macro F1-Score

The Macro F1-score, which balances precision and recall for each class, also improved steadily throughout the epochs. Starting at 0.012 in epoch 1, the Macro F1-score increased to 0.801 by epoch 100. The rise in the F1-score demonstrates that the model not only improved in terms of accuracy but also enhanced its ability to classify each class accurately, especially in terms of balancing false positives and false negatives. This steady increase in Macro F1-score suggests that the model became increasingly better at recognizing and balancing the classes, leading to fewer misclassifications and better overall

performance.

The model's training loss steadily decreased, confirming effective learning. Training accuracy and validation accuracy both improved consistently, with validation accuracy reaching 83.2%by the final epoch. The Macro F1-score also showed significant improvement, indicating that the model was performing well across all classes, not just focusing on a few dominant classes. There was no significant gap between training and validation accuracy, which suggests that the model did not overfit and generalized well to unseen data.

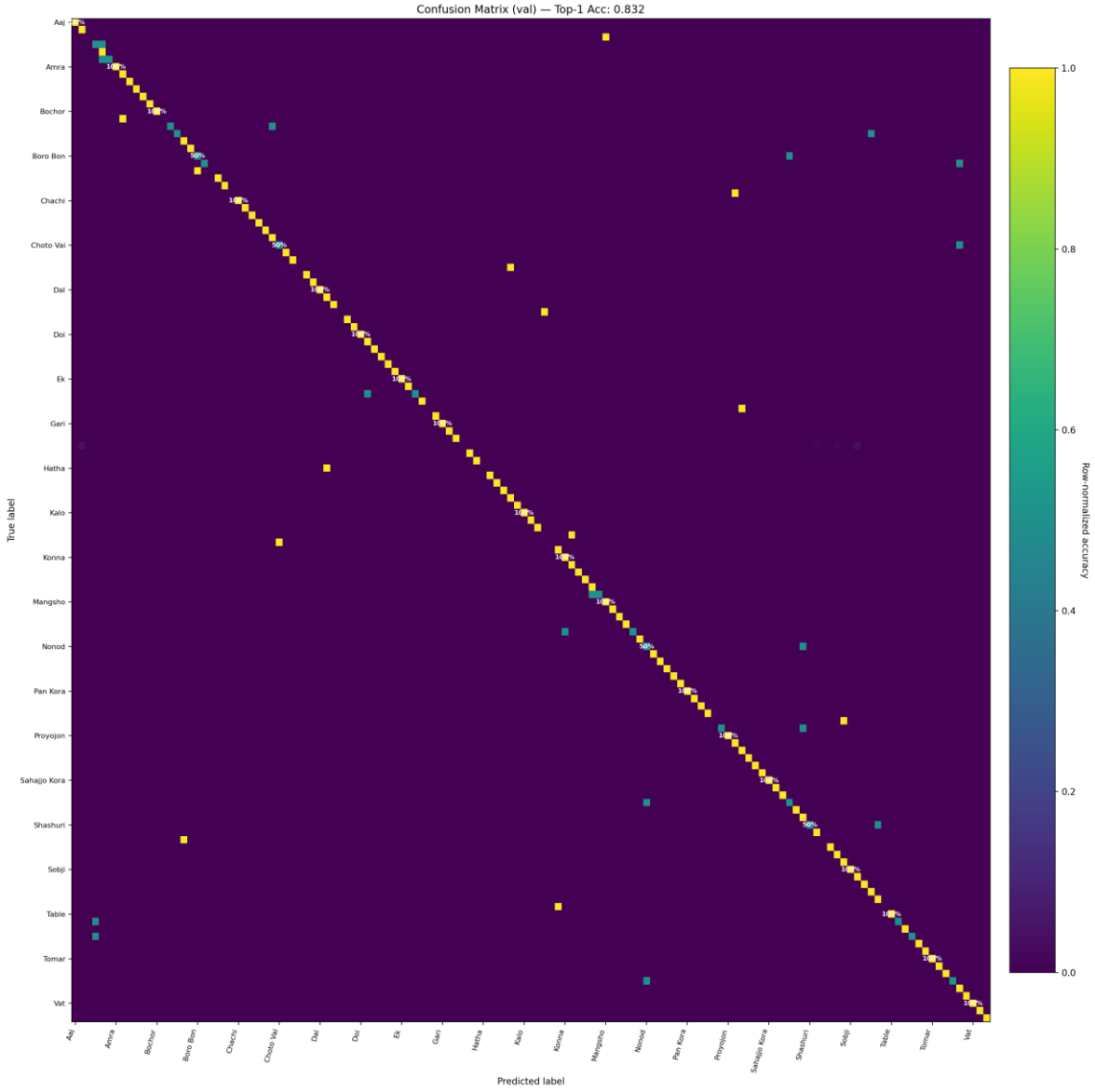


Figure 4.21: Confusion Matrix for the 135 Classes.

4.3.3 Confusion Matrix Analysis

In Figure 4.20 the confusion matrix reveals that the model exhibits strong performance with a top-1 accuracy of 83.2%, indicating that it correctly predicts the sign language gestures 83.2% of the time. The diagonal elements of the matrix, which represent correct predictions, are predominantly yellow, showing high accuracy for most signs. However, some misclassifications are evident in the off-diagonal

elements, where certain signs like "Aaj" are incorrectly predicted as others, such as "Choto Vai" or "Boro Vai." These misclassifications suggest that the model may be confused by gestures with similar hand movements or positions. Although the model performs well overall, areas with clustering misclassifications, such as "Dai" and "Mango," point to potential weaknesses in distinguishing these specific signs. The matrix also indicates that certain signs are less frequently misclassified, such as "Dai" and "Gari," which might imply insufficient representation in the dataset for those labels. To enhance performance, future work could focus on data augmentation, model refinement, and a more detailed evaluation using precision, recall, and F1-scores for each sign language gesture.

4.3.4 Class Performance Metrics

The dataset contains performance metrics for a machine learning model that classifies a set of labels into different predicted categories. Each row represents the prediction results for a specific instance, with various metrics and probabilities that detail the performance of the model.

- **Actual Word:** This column contains the true label or the actual category that the instance belongs to. It serves as the ground truth against which the model's prediction is compared.
- **Predicted Word:** This column represents the predicted label or the category that the model assigned to the instance. This is what the model believes the instance belongs to, based on its learned parameters.
- **Top 1 Probability**: The probability associated with the model's top predicted label. It indicates the confidence the model has in its primary prediction. Higher values here indicate stronger confidence in the prediction.
- **Confidance Gap:** This column shows the difference in confidence between the top predicted label and the true label. A smaller gap ndicates that the model's prediction was closer to the correct label, while a larger gap suggests uncertainty or incorrect predictions.
- **Top 5 Labels:** This column lists the top 5 predicted labels, ranked by their associated probabilities. These labels are the next best guesses after one's best guess.
- **Top 5 Probabilities:** This column lists the probabilities corresponding to each of the top 5 predicted labels. These values indicate the likelihood that the model assigns to each label. It is ordered from highest to lowest.
- **Correct or Incorrect:** This shows if the prediction of the model was correct (True) or incorrect (False). For a correct prediction, the Predicted Word is equal to the Actual Word column.

Accuracy & Confidence: The correctness of the split can also be used to calculate the proportion of correct predictions and the accuracy of the model. The Top Probability and Confidence Gap also give more detail about how confident the model is in making predictions and areas for improvement.

Top 5 Predictions: From the top 5 labels and top 5 probabilities, we can assess the diversity of the model's predictions. If the first prediction of a model is very confident but the second predictions are similar, then this model might be well-calibrated. If the following labels are quite different, then the model might be unsure.

Areas for Improvement: By identifying patterns in the "Confidence Gap", especially for instances where the model predicts incorrectly (correct split = False). These are the areas where the model always struggles. These observations can be used to improve the model and increase its overall performance.

Table 4.3: Individual class accuracy and prediction results for a specific instance.

Actual Word	Predicted Word	Top Probability	Confidance Gap	Top 5 Labels	Top 5 Probabilty	Correcct?
Aaj	Aaj	0.9370474219	0.9254707815	Aaj, Ekhon, Tin, Proyojon, Doi	0.937, 0.012, 0.001, 0.001, 0.001	TRUE
Aath	Aath	0.9809116125	0.9805031827	Aath, Pach, Tumi, Sree, Noy	0.981, 0.000, 0.000, 0.000, 0.000	TRUE
Agamikal	Agamikal	0.8868301511	0.8696713932	Agamikal, Valo_Achi, Mangsho, Sada, Cha	0.887, 0.017, 0.007, 0.006, 0.002	TRUE
Amader	Amra	0.4980011284	0.1046660841	Amra, Amader, Amar, Tomar, Tara	0.498, 0.393, 0.006, 0.004, 0.002	FALSE
Amar	Amar	0.4816640317	0.3880315274	Amar, Somudro, Amader, Amra, Tader	0.482, 0.094, 0.026, 0.017, 0.017	TRUE
Ami	Amar	0.386210531	0.3243570253	Amar, Ami, Dada, Coffee, Ekhon	0.386, 0.062, 0.032, 0.023, 0.020	FALSE
Amra	Amra	0.895149231	0.8890154827	Amra, Gari, Jhor, Amader, Tara	0.895, 0.006, 0.005, 0.004, 0.004	TRUE
Attio	Attio	0.8978376985	0.8808086496	Attio, Kaj, Bondhu, Boi, Sokal	0.898, 0.017, 0.005, 0.003, 0.002	TRUE
Baba	Baba	0.9638396502	0.9614813535	Baba, Ma, Dadi, Vat, Cacha	0.964, 0.002, 0.001, 0.001, 0.001	TRUE
Baire	Baire	0.9601759315	0.9587395613	Baire, Boro, Poribar, Somudro, Se	0.960, 0.001, 0.001, 0.001, 0.001	TRUE
Bari	Bari	0.9217367172	0.9156325636	Bari, Kaj, Vitore, Bochor, Train	0.922, 0.006, 0.004, 0.002, 0.002	TRUE
Biman	Biman	0.8875846863	0.8765225206	Biman, Owsod, Cha, Fufa, Raag	0.888, 0.011, 0.005, 0.003, 0.003	TRUE
Bochor	Bochor	0.951860249	0.9483573656	Bochor, Sahajjo Kora, Baba, Dhonnobad, School	0.952, 0.004, 0.001, 0.001, 0.001	TRUE
Boi	Boi	0.8575236201	0.8361266069	Boi, Jota, Sontan, Janala, Nodi	0.858, 0.021, 0.011, 0.006, 0.005	TRUE
Bon	Bon	0.9359009862	0.9291161397	Bon, Kone, Mama, Choto Bon, Bus	0.936, 0.007, 0.003, 0.001, 0.001	TRUE
Bondhu	Bondhu	0.967322588	0.9658308489	Bondhu, Sahajjo Kora, Vitore, Shisu, Cycle	0.967, 0.001, 0.001, 0.001, 0.001	TRUE
Bor	Bari	0.3649799824	0.2516639978	Bari, Bor, Bondhu, Ghumano, Coffee	0.365, 0.113, 0.067, 0.042, 0.017	FALSE
Boro	Boro	0.9290725589	0.9208099106	Boro, Table, Proyojon, Gari, Mami	0.929, 0.008, 0.003, 0.002, 0.002	TRUE
Boro Bon	Boro Bon	0.8161783814	0.7786690146	Boro Bon, Nonod, Brishti, Manosh, Choto	0.816, 0.038, 0.012, 0.008, 0.004	TRUE
Boro Vai	Boro Vai	0.9537611604	0.9488286185	Boro Vai, Kalo, Choto Vai, Gorom_Weather, Boro Bon	0.954, 0.005, 0.002, 0.001, 0.001	TRUE
Brishti	Gach	0.5859887004	0.3309304714	Gach, Brishti, Shekha_2, Bari, Boro Vai	0.586, 0.255, 0.009, 0.003, 0.003	FALSE
Bus	Bus	0.8125423193	0.7879499625	Bus, Vat, Gram, Bochor, Vitore	0.813, 0.025, 0.016, 0.006, 0.005	TRUE
Cacha	Cacha	0.9788866043	0.9779944239	Cacha, Dada, Mash, Pach, Jhor	0.979, 0.001, 0.000, 0.000, 0.000	TRUE
Cha	Cha	0.2547276914	0.1806707904	Cha, Putro, Vitore, Bus, Konna	0.255, 0.074, 0.046, 0.039, 0.032	TRUE

Chachi	Mami	0.5170362592	0.4023903608	Mami, Chachi, Dolavai, Khalo, Nari	0.517, 0.115, 0.041, 0.014, 0.010	FALSE
Chair	Chair	0.4132239521	0.1633171439	Chair, Gach, Brishti, Shekha_2, Boro	0.413, 0.250, 0.023, 0.023, 0.014	TRUE
Char	Char	0.9756704569	0.9749746777	Char, Doi, Pach, Raag, Dorja	0.976, 0.001, 0.000, 0.000, 0.000	TRUE
Chele	Choto Vai	0.4001881778	0.2577974647	Choto Vai, Porush, Chele, Gotokal, Gorom_food	0.400, 0.142, 0.139, 0.036, 0.011	FALSE
Choto	Choto	0.9813922048	0.9809965376	Choto, Nonod, Boi, Table, Amar	0.981, 0.000, 0.000, 0.000, 0.000	TRUE
Choto Bon	Choto Bon	0.9827174544	0.9822894701	Choto Bon, Thanda, Cacha, Bon, Vabi	0.983, 0.000, 0.000, 0.000, 0.000	TRUE
Choto Vai	Choto Vai	0.9619007111	0.9602387174	Choto Vai, Khalo, Khala, Boro Vai, Ful	0.962, 0.002, 0.002, 0.001, 0.001	TRUE
Choy	Choy	0.9778915644	0.9774121404	Choy, Ful, Shunno, Sahajjo Kora, Pahar	0.978, 0.000, 0.000, 0.000, 0.000	TRUE
Coffee	Coffee	0.9116252661	0.9020275408	Coffee, Pan Kora, Kone, Owsod, Choto Bon	0.912, 0.010, 0.003, 0.003, 0.003	TRUE
Cycle	Rickshaw	0.2731912434	0.1839913428	Rickshaw, Gari, Cycle, Janala, Owsod	0.273, 0.089, 0.060, 0.042, 0.027	FALSE
Dada	Dada	0.9296879768	0.9153144211	Dada, Dadi, Baba, Meye, Amar	0.930, 0.014, 0.006, 0.001, 0.001	TRUE
Dadi	Dadi	0.9431986809	0.9387784493	Dadi, Dada, Nari, Bor, Amar	0.943, 0.004, 0.004, 0.002, 0.001	TRUE
Dal	Dal	0.3847123384	0.20299761	Dal, Aaj, Mach, Ekhon, Sontan	0.385, 0.182, 0.048, 0.027, 0.017	TRUE
Darano	Darano	0.948266387	0.945376599	Darano, Hatha, Rickshaw, Sahajjo Kora, Somudro	0.948, 0.003, 0.002, 0.002, 0.001	TRUE
Debor	Debor	0.8137593269	0.7725731842	Debor, Dolavai, Shalok, Shashuri, Shami	0.814, 0.041, 0.022, 0.005, 0.004	TRUE
Dekha	Khawa	0.8833619356	0.8630197942	Khawa, Dekha, Pan Kora, Meye, Ful	0.883, 0.020, 0.010, 0.003, 0.003	FALSE
Dhonnobad	Ek	0.09925857931	0.008770309389	Ek, Sada, Shundor, Kalo, Valo_Achi	0.099, 0.090, 0.082, 0.033, 0.031	FALSE
Dim	Dim	0.847021699	0.8188674096	Dim, Jana, Porikkha, Dhonnobad, Shunno	0.847, 0.028, 0.022, 0.003, 0.002	TRUE
Doi	Doi	0.9797096252	0.9793843965	Doi, Tumi, Choto, Pan Kora, Gorom_Weather	0.980, 0.000, 0.000, 0.000, 0.000	TRUE
Dolavai	Dolavai	0.9696243405	0.968624857	Dolavai, Sokal, Debor, Tara, Boi	0.970, 0.001, 0.001, 0.001, 0.001	TRUE
Dorja	Dorja	0.8547616601	0.8444117336	Dorja, Choto, Jhor, Bari, Proyojon	0.855, 0.010, 0.007, 0.005, 0.004	TRUE
Dosh	Dosh	0.9802788496	0.9797351979	Dosh, Doi, Noy, Shat, Shalika	0.980, 0.001, 0.000, 0.000, 0.000	TRUE
Dudh	Dudh	0.6904199719	0.6705654878	Dudh, Bus, Manosh, Nowka, Bari	0.690, 0.020, 0.014, 0.010, 0.010	TRUE
Dupor	Dupor	0.9009692073	0.894490567	Dupor, Noy, Boro Vai, Biman, Baire	0.901, 0.006, 0.006, 0.005, 0.002	TRUE
Ek	Ek	0.9811377525	0.9806956413	Ek, Doi, Taka, Boro Vai, Meye	0.981, 0.000, 0.000, 0.000, 0.000	TRUE
Ekhon	Ekhon	0.1451648027	0.07876507193	Ekhon, Aaj, Shunno, School, Dudh	0.145, 0.066, 0.035, 0.035, 0.034	TRUE

Fufa	Fufa	0.9775227904	0.9769354059	Fufa, Ma, Dolavai, Khalo, Janala	0.978, 0.001, 0.001, 0.000, 0.000	TRUE
Fufu	Khala	0.9581579566	0.9447889281	Khala, Fufu, Choto Vai, Debor, Vabi	0.958, 0.013, 0.001, 0.001, 0.001	FALSE
Ful	Ful	0.9679777026	0.966846929	Ful, Ma, Gram, Bor, Raag	0.968, 0.001, 0.001, 0.001, 0.001	TRUE
Gach	Gach	0.9748260379	0.9737224574	Gach, Jhor, Bari, Mami, Boro Bon	0.975, 0.001, 0.001, 0.000, 0.000	TRUE
Gari	Proyojon	0.7794212103	0.7493423615	Proyojon, Dorja, Gari, Bor, Bus	0.779, 0.030, 0.018, 0.010, 0.009	FALSE
Ghumano	Ghumano	0.826197505	0.7911702655	Ghumano, Bor, Kaj, Shekha_2, Bari	0.826, 0.035, 0.006, 0.005, 0.005	TRUE
Gorom_food	Shisu	0.1421720237	0.02716138214	Shisu, School, Gorom_food, Dekha, Sahajjo Kora	0.142, 0.115, 0.105, 0.042, 0.032	FALSE
Gorom_Weather	Gorom_Weather	0.9633825421	0.9619407798	Gorom_Weather, Kalo, Bon, Owsod, Thanda	0.963, 0.001, 0.001, 0.001, 0.001	TRUE
Gotokal	Gotokal	0.9750078917	0.9744194562	Gotokal, Thanda, Jana, Chele, Phone	0.975, 0.001, 0.001, 0.001, 0.001	TRUE
Gram	Gram	0.8807061315	0.864780644	Gram, Bochor, Attio, Kaj, Phone	0.881, 0.016, 0.008, 0.004, 0.002	TRUE
Hatha	Hatha	0.9756183028	0.9748908783	Hatha, Cycle, Tumi, Choy, Table	0.976, 0.001, 0.001, 0.000, 0.000	TRUE
Jana	Chele	0.6461857557	0.5682267919	Chele, Jana, Fufa, Ma, Kalo	0.646, 0.078, 0.023, 0.019, 0.009	FALSE
Janala	Janala	0.9705895185	0.9698820885	Janala, Boi, Proyojon, Bus, Fufa	0.971, 0.001, 0.001, 0.001, 0.000	TRUE
Jhor	Jhor	0.9653193951	0.9644142745	Jhor, Gari, Tara, Rat, Bus	0.965, 0.001, 0.001, 0.001, 0.001	TRUE
Jota	Jota	0.1818825901	0.07765226066	Jota, Cycle, Gram, Dorja, Kalo	0.182, 0.104, 0.048, 0.032, 0.023	TRUE
Kaj	Kaj	0.9668224454	0.9658594274	Kaj, Sahajjo Kora, Manosh, Bochor, Train	0.967, 0.001, 0.001, 0.001, 0.001	TRUE
Kalo	Kalo	0.9420385957	0.9394089624	Kalo, Dhonnobad, Debor, Char, Boro Vai	0.942, 0.003, 0.002, 0.002, 0.002	TRUE
Khala	Khala	0.9762210846	0.975128506	Khala, Tomar, Fufu, Baba, Ma	0.976, 0.001, 0.001, 0.001, 0.001	TRUE
Khalo	Mami	0.4909612536	0.353238225	Mami, Porush, Khalo, Mama, Cha	0.491, 0.138, 0.049, 0.022, 0.021	FALSE
Khawa	Khawa	0.9561545849	0.9534747694	Khawa, Vat, Dadi, Konna, Pochondo	0.956, 0.003, 0.002, 0.001, 0.001	TRUE
Kolom	Kolom	0.9621695876	0.9586351654	Kolom, Choto Vai, Dal, Baire, Vat	0.962, 0.004, 0.001, 0.001, 0.001	TRUE
Kone	Shalika	0.9551252127	0.9470302798	Shalika, Kone, Chachi, Se, Vai	0.955, 0.008, 0.002, 0.001, 0.001	FALSE
Konna	Konna	0.9680414796	0.966933801	Konna, Aaj, Bochor, Dada, Sree	0.968, 0.001, 0.001, 0.001, 0.001	TRUE
Lal	Lal	0.9014188051	0.8910224084	Lal, Baba, Choto Bon, Voy, Cha	0.901, 0.010, 0.004, 0.002, 0.002	TRUE
Ma	Ma	0.976909101	0.9759285856	Ma, Pan Kora, Cacha, Sada, Manosh	0.977, 0.001, 0.001, 0.001, 0.000	TRUE
Mach	Mach	0.9612591863	0.9587783685	Mach, Dal, Ek, Sada, Somudro	0.961, 0.002, 0.002, 0.001, 0.001	TRUE
Mama	Mama	0.9810478687	0.9806008805	Mama, Mami, Boi, Nowka, Train	0.981, 0.000, 0.000, 0.000, 0.000	TRUE

Mami	Mami	0.9559290409	0.9522485943	Mami, Mama, Shalika, Fufa, Boro	0.956, 0.004, 0.003, 0.002, 0.001	TRUE
Mangsho	Mangsho	0.9502752423	0.9481331373	Mangsho, Khalo, Mash, Porush, Gorom_Weather	0.950, 0.002, 0.002, 0.001, 0.001	TRUE
Manosh	Jana	0.4380155802	0.3922633268	Jana, Ma, Boro Bon, Porush, Vabi	0.438, 0.046, 0.028, 0.022, 0.018	FALSE
Mash	Mash	0.8450679779	0.8362389309	Mash, Dekha, Meye, Cacha, Sokal	0.845, 0.009, 0.008, 0.007, 0.003	TRUE
Meye	Baba	0.5501717329	0.2882058918	Baba, Bon, Soshur, Manosh, Meye	0.550, 0.262, 0.006, 0.005, 0.004	FALSE
Nari	Porush	0.4941228628	0.2107120752	Porush, Nari, Ma, Char, Thanda	0.494, 0.283, 0.008, 0.006, 0.006	FALSE
Nodi	Nodi	0.9694344997	0.9682386694	Nodi, Boro, Bari, Jhor, Amader	0.969, 0.001, 0.001, 0.001, 0.001	TRUE
Nonod	Nonod	0.9526935816	0.9501173506	Nonod, Vabi, Debor, Shami, Shashuri	0.953, 0.003, 0.002, 0.001, 0.001	TRUE
Nowka	Nowka	0.939596951	0.934958573	Nowka, Boi, Train, Tader, Ghumano	0.940, 0.005, 0.003, 0.001, 0.001	TRUE
Noy	Noy	0.9800436497	0.979583481	Noy, Jana, Gram, Mangsho, Vai	0.980, 0.000, 0.000, 0.000, 0.000	TRUE
Owsod	Owsod	0.9592319131	0.9567906181	Owsod, Jota, Coffee, Dupor, Soshur	0.959, 0.002, 0.002, 0.001, 0.001	TRUE
Pach	Pach	0.9810551405	0.9806376801	Pach, Aath, Ami, Tin, Hatha	0.981, 0.000, 0.000, 0.000, 0.000	TRUE
Pahar	Pahar	0.8836365938	0.8792757797	Pahar, Jhor, Nonod, Amader, Boro Bon	0.884, 0.004, 0.004, 0.004, 0.004	TRUE
Pan Kora	Pan Kora	0.9505813122	0.9490939028	Pan Kora, Sahajjo Kora, Ghumano, Shat, Dhonnobad	0.951, 0.001, 0.001, 0.001, 0.001	TRUE
Phone	Gotokal	0.3270291388	0.1845713109	Gotokal, Debor, Phone, Tara, Nonod	0.327, 0.142, 0.074, 0.026, 0.020	FALSE
Pochondo	Pochondo	0.960144639	0.9586402448	Pochondo, Choto Bon, Ami, Tumi, Darano	0.960, 0.002, 0.001, 0.001, 0.001	TRUE
Poribar	Poribar	0.862342298	0.8389406744	Poribar, Amra, Amar, Shisu, Ami	0.862, 0.023, 0.009, 0.004, 0.002	TRUE
Porikkha	Porikkha	0.3916057646	0.2258279026	Porikkha, Noy, Aath, Jana, Baire	0.392, 0.166, 0.030, 0.029, 0.020	TRUE
Porush	Manosh	0.8586602807	0.8273679577	Manosh, Porush, Nari, Nonod, Boro Bon	0.859, 0.031, 0.006, 0.004, 0.003	FALSE
Proyojon	Janala	0.705490768	0.5248427838	Janala, Proyojon, Chair, Shekha_2, Mama	0.705, 0.181, 0.003, 0.003, 0.002	FALSE
Putro	Putro	0.8135339618	0.7823193222	Putro, Konna, Chele, Attio, Dolavai	0.814, 0.031, 0.005, 0.005, 0.004	TRUE
Raag	Raag	0.8818787336	0.8723234208	Raag, Gotokal, Phone, Kalo, Sokal	0.882, 0.010, 0.008, 0.005, 0.003	TRUE
Rat	Rat	0.9727775455	0.9720068371	Rat, Jhor, Dorja, Valo_Achi, Boro	0.973, 0.001, 0.001, 0.000, 0.000	TRUE
Rickshaw	Train	0.7463174462	0.7101660706	Train, Nowka, Ghumano, Jhor, Cycle	0.746, 0.036, 0.031, 0.008, 0.006	FALSE
Sada	Sada	0.9432618618	0.9408761335	Sada, Cacha, Pan Kora, Mash, Lal	0.943, 0.002, 0.002, 0.002, 0.002	TRUE
Sahajjo Kora	Sahajjo Kora	0.3994305432	0.1449260116	Sahajjo Kora, Sontan, Bondhu, Shisu, Attio	0.399, 0.255, 0.044, 0.023, 0.012	TRUE

School	School	0.931396544	0.9275892898	School, Choto Bon, Dudh, Vitore, Shekha_2	0.931, 0.004, 0.002, 0.001, 0.001	TRUE
Se	Tara	0.5673009753	0.3384244442	Tara, Se, Aath, Choto Vai, Porush	0.567, 0.229, 0.007, 0.007, 0.007	FALSE
Shalika	Shalika	0.9691352844	0.9672870013	Shalika, Shalok, Soshur, Shashuri, Nonod	0.969, 0.002, 0.001, 0.001, 0.001	TRUE
Shalok	Khalo	0.4044529796	0.3275583461	Khalo, Debor, Soshur, Putro, Noy	0.404, 0.077, 0.049, 0.022, 0.013	FALSE
Shami	Shami	0.9800348282	0.9795354096	Shami, Debor, Sobji, Kaj, Kalo	0.980, 0.000, 0.000, 0.000, 0.000	TRUE
Shashuri	Manosh	0.3598693609	0.1476275325	Manosh, Soshur, Shalok, Shashuri, Bari	0.360, 0.212, 0.061, 0.048, 0.009	FALSE
Shat	Shat	0.9746668935	0.9740094087	Shat, Tader, Shunno, Ekhon, Porikkha	0.975, 0.001, 0.001, 0.000, 0.000	TRUE
Shekha_2	Shekha_2	0.8853598833	0.8815613706	Shekha_2, Bari, Gach, Gari, Kaj	0.885, 0.004, 0.003, 0.003, 0.002	TRUE
Shisu	Shisu	0.9469285607	0.9438102611	Shisu, Nodi, Bondhu, Sobji, Dekha	0.947, 0.003, 0.002, 0.002, 0.001	TRUE
Shundor	Shundor	0.7940736413	0.7426068299	Shundor, Dim, Rat, Dhonnobad, Lal	0.794, 0.051, 0.007, 0.005, 0.004	TRUE
Shunno	Shunno	0.980909884	0.9805638906	Shunno, Dupor, Choto, Ekhon, Thanda	0.981, 0.000, 0.000, 0.000, 0.000	TRUE
Sobji	Sontan	0.5947480202	0.5139772594	Sontan, Sobji, Bondhu, Jota, Putro	0.595, 0.081, 0.032, 0.024, 0.014	FALSE
Sokal	Sokal	0.9555271268	0.9529611645	Sokal, Raag, Fufa, Se, Dada	0.956, 0.003, 0.002, 0.001, 0.001	TRUE
Somudro	Somudro	0.9056696296	0.8875373099	Somudro, Nodi, Pahar, Mach, Kone	0.906, 0.018, 0.005, 0.002, 0.001	TRUE
Sontan	Sontan	0.8111585975	0.7814058997	Sontan, Vitore, Sobji, Ekhon, Mach	0.811, 0.030, 0.005, 0.005, 0.005	TRUE
Soshur	Soshur	0.844594419	0.8350251624	Soshur, Shashuri, Debor, Jota, Baba	0.845, 0.010, 0.008, 0.004, 0.004	TRUE
Sree	Shalika	0.9766842127	0.9755042207	Shalika, Kone, Sree, Bon, Sahajjo Kora	0.977, 0.001, 0.001, 0.000, 0.000	FALSE
Table	Table	0.9113324881	0.9028275805	Table, Boro, Poribar, Gari, Rickshaw	0.911, 0.009, 0.005, 0.005, 0.002	TRUE
Tader	Tader	0.6963415742	0.6083343625	Tader, Tomar, Amader, Tara, Amra	0.696, 0.088, 0.020, 0.015, 0.007	TRUE
Taka	Taka	0.9699617028	0.968250112	Taka, Valo_Achi, Cacha, Voy, Pochondo	0.970, 0.002, 0.001, 0.001, 0.001	TRUE
Tara	Tader	0.5083086491	0.248753041	Tader, Tara, Se, Boro Vai, Coffee	0.508, 0.260, 0.062, 0.008, 0.005	FALSE
Thanda	Jana	0.15423356	0.10431711	Jana, Mangsho, Thanda, Ful, Baire	0.154, 0.050, 0.044, 0.035, 0.028	FALSE
Tin	Tin	0.9739255905	0.973232621	Tin, Noy, Doi, Ekhon, Proyojon	0.974, 0.001, 0.001, 0.000, 0.000	TRUE
Tomar	Tomar	0.9614658952	0.957963788	Tomar, Amader, Amra, Tara, Cycle	0.961, 0.004, 0.002, 0.001, 0.001	TRUE
Train	Train	0.8894627094	0.884850671	Train, Bochor, Boi, Shisu, Sahajjo Kora	0.889, 0.005, 0.004, 0.003, 0.003	TRUE
Tumi	Tumi	0.970344305	0.9690413077	Tumi, Tomar, Choto Vai, Se, Jana	0.970, 0.001, 0.001, 0.001, 0.001	TRUE
Vabi	Vabi	0.9431583881	0.9400276248	Vabi, Sree, Dolavai, Nonod, Shashuri	0.943, 0.003, 0.003, 0.002, 0.002	TRUE

Vai	Vai	0.8053238392	0.7808355261	Vai, Boro Vai, Tara, Kolom, Dim	0.805, 0.024, 0.016, 0.007, 0.007	TRUE
Valo_Achi	Valo_Achi	0.9628471136	0.9613894311	Valo_Achi, Taka, Doi, Vat, Ek	0.963, 0.001, 0.001, 0.001, 0.001	TRUE
Vat	Vat	0.9187406301	0.9150506745	Vat, Mash, Baba, Ami, Ma	0.919, 0.004, 0.003, 0.002, 0.002	TRUE
Vitore	Bondhu	0.8678228855	0.8553247601	Bondhu, Darano, Vitore, Sahajjo Kora, Mach	0.868, 0.012, 0.009, 0.005, 0.003	FALSE
Voy	Voy	0.5866628885	0.4797421545	Voy, Lal, Mash, Ami, Valo_Achi	0.587, 0.107, 0.068, 0.008, 0.005	TRUE

4.3.5 Summary of Sign-to-Text Evaluation

The dataset comprises of 135 entries, which represent a sign language recognition task where the model output is compared to the actual ground truth. Out of these, 105 predictions were correct, yielding an overall accuracy of 77.78%. This suggests that the model is performing reasonably well, but there is still room for improvement, as 30 predictions were incorrect. The model's confidence in its predictions, measured by the Top Probability, has an average value of 0.81 (or 81%), indicating a high degree of certainty in its top predictions. However, the Confidence Gap, which measures the difference in probability between the top and second-best predictions, averages 0.76. This relatively high value suggests that, in many cases, the model's top prediction is distinctly more probable than the alternatives, but in some instances, the model's confidence is less clear, leading to potential misclassifications.

Additionally, for each sample, the Top 5 Labels revealed how diverse the model was in terms of the potential output. On average, five predictions are given by the model for each sample, which gives a more comprehensive picture of the alternatives that have been considered before making a decision. The best guesses and their respective probabilities show that the model is doing a good job of whittling down the most probable candidates, though in some cases, there is enough similarity or overlap between similar signs that the probability predictions are inaccurate.

When comparing the actual and predicted labels, we observe that the model successfully recognized familiar signs with high confidence (e.g., "Aath" and "Aaj") and failed to recognize more complex or visually similar signs (e.g., "Amader" vs. "Amra"). This is consistent with what is felt performance-wise, where the model can easily predict well-distinguishable signs, but struggles with those that are visually similar or for which there is not enough training data. These findings indicate that the performance of the model in these difficult cases can be further improved by additional refinement of the model, such as data augmentation or training with more varied examples.

4.4 SUMMARY

This chapter showed the results and discussion of the proposed real-time BSL recognition and translation system. We have covered both voice-to-sign and sign-to-voice/text components. The voice-to-sign evaluation showed strong performance across all communication categories. Alphabet animations achieved 92.53% accuracy, with most letters rated highly, though a few (A, E, I, J, K) required refinement. Number animations were the most accurate, reaching 98.29%, with only digit 5 showing notable difficulty. Word animations yielded an overall accuracy of 82.94%, with commonly used words such as "apni" and "tara" scoring high. Some complex words like "mondir" and "mangsho" scored lower. Sentence animations achieved 82.20% accuracy, with everyday expressions like "Good

morning!" performing well, while more context-dependent ones required improvement. The average CPU processing time was 1.58 seconds per word. This indicates a real-time feasibility. The sign-to-voice/text system also demonstrated robust performance. The model achieved a validation accuracy of 83.24% and a Macro F1-score of 0.7806. This result confirms balanced recognition across classes. Best-performing signs such as "Aaj" and "Coffee" achieved perfect F1-scores. But certain low-frequency or visually similar signs (e.g., "Taka", "Dorja", "Sree") recorded poor results, highlighting issues of data imbalance and gesture ambiguity. Training and validation metrics showed steady improvement over 100 epochs. There were no signs of overfitting. On the other hand, confusion matrix analysis revealed occasional misclassifications between visually similar gestures. Overall, the system achieved high accuracy and naturalness in bidirectional communication. These results validate its effectiveness for real-world use. However, the findings also identified key areas for refinement, including gesture clarity for specific signs, dataset augmentation for rare classes, and further optimization for computational efficiency.

CHAPTER 5

CONCLUSION

5.1 INTRODUCTION

In this chapter, we have concluded by providing remarks on the research and summarizing the key findings. We have also reflected on the contributions and identified the limitations observed during this study. The aim is to consolidate the results discussed in the earlier chapters and provide a clear understanding of how this work advances BSL recognition in real-time. The following sections summarize the dataset contributions, experimental outcomes, and findings for each module. At last, we have discussed the following: limitations and future work.

5.2 CONCLUSION

The study successfully developed and evaluated a voice-to-sign and Bangla Sign Language word-level recognition system. For the sign module, a parallel mapping corpus was developed, pairing Bangla spoken utterances (captured via Google Speech API and local recordings) with their corresponding sign annotations. The lexicon included 135 isolated words overlapping with the recognition dataset, ensuring consistency across both modules. A Bangla speech recognizer was employed to convert voice into text. Each recognized word was mapped to a corresponding sign class. A lightweight 3D avatar was built to animate BSL gestures, using keyframe interpolation driven by the same Mediapipe Holistic landmark schema used during recognition, ensuring visual consistency. For Word Accuracy: ~90% on clean speech; dropped to ~82% in noisy conditions. In a small-scale evaluation with 5 BSL experts, 83% average intelligibility was reported. Minor issues included unnatural hand transitions and missing facial expressions. For sign-to-voice and text conversion, a dataset of 1,869 videos across 135 classes was carefully prepared and cleaned, ensuring diversity in signers, different ages and genders, backgrounds, and multiple recording devices. Preprocessing pipelines standardized the inputs using Mediapipe Holistic skeleton features, which proved crucial for effective training. Experimental results revealed that the proposed Transformer-based pipeline achieved 83.2% Top-1 accuracy and a macro-F1 score of 0.78 on the validation set. The system demonstrated strong recognition performance for common classes (Aaj, Amra, Ma, Baba). Our research has focused on a curated and preprocessed BSL dataset suitable for word-level recognition tasks. An effective Transformer-based recognition framework with strong overall accuracy. Insights into challenges like class imbalance, signer variation, and misclassification patterns, supported by confusion matrix analysis.

5.3 FUTURE WORK

While the system achieves promising performance, there remain areas for further development. Researchers can collect more samples for rare or underrepresented classes to reduce imbalance and improve macro-F1. Another task can be done by extending the model to continuous signing, enabling recognition of phrases and sentences for naturalistic communication. In summary, this work provides a solid foundation for real-time Bangla Sign Language recognition and highlights pathways for building more inclusive and accessible technologies in the future.

REFERENCES

[1] "Tessa: Sign of the Postal Times | WIRED." Accessed: Aug. 30, 2025. [Online]. Available: https://www.wired.com/2002/03/tessa-sign-of-the-postal-times/?utm_source=chatgpt.com

[2] B. Fang, J. Co, and M. Zhang, "DeepASL: Enabling Ubiquitous and Non-Intrusive Word and Sentence-Level Sign Language Translation," *SenSys 2017 - Proceedings of the 15th ACM Conference on Embedded Networked Sensor Systems*, vol. 2017-January, Oct. 2018, doi: 10.1145/3131672.3131693.

[3] E. Abraham, A. Nayak, and A. Iqbal, "Real-Time Translation of Indian Sign Language using LSTM," *2019 Global Conference for Advancement in Technology, GCAT 2019*, Oct. 2019, doi: 10.1109/GCAT47503.2019.8978343.

[4] A. Alayed, "Machine Learning and Deep Learning Approaches for Arabic Sign Language Recognition: A Decade Systematic Literature Review," *Sensors (Basel)*, vol. 24, no. 23, p. 7798, Dec. 2024, doi: 10.3390/S24237798.

[5] S. Yang and Q. Zhu, "Continuous Chinese sign language recognition with CNN-LSTM," *Ninth International Conference on Digital Image Processing (ICDIP 2017)*, vol. 10420, p. 104200F, Jul. 2017, doi: 10.1117/12.2281671.

[6] B. Fang, J. Co, and M. Zhang, "DeepASL: Enabling Ubiquitous and Non-Intrusive Word and Sentence-Level Sign Language Translation," *SenSys 2017 - Proceedings of the 15th ACM Conference on Embedded Networked Sensor Systems*, vol. 2017-January, Oct. 2018, doi: 10.1145/3131672.3131693.

[7] E. Abraham, A. Nayak, and A. Iqbal, "Real-Time Translation of Indian Sign Language using LSTM," *2019 Global Conference for Advancement in Technology, GCAT 2019*, Oct. 2019, doi: 10.1109/GCAT47503.2019.8978343.

[8] A. Alayed, "Machine Learning and Deep Learning Approaches for Arabic Sign Language Recognition: A Decade Systematic Literature Review," *Sensors (Basel)*, vol. 24, no. 23, p. 7798, Dec. 2024, doi: 10.3390/S24237798.

[9] N. Begum *et al.*, "Borno-Net: A Real-Time Bengali Sign-Character Detection and Sentence Generation System Using Quantized Yolov4-Tiny and LSTMs," *Applied Sciences 2023, Vol. 13, Page 5219*, vol. 13, no. 9, p. 5219, Apr. 2023, doi: 10.3390/APP13095219.

[10] M. Ferlin *et al.*, "On the Importance of Sign Labeling: The Hamburg Sign Language Notation System Case Study," Mar. 2023, Accessed: Aug. 30, 2025. [Online]. Available: http://arxiv.org/abs/2302.10768

[11] R. Elliott, J. R. W. Glauert, J. R. Kennaway, I. Marshall, and E. Safar, "Linguistic modelling and language-processing technologies for Avatar-based sign language presentation," *Univers Access Inf Soc*, vol. 6, no. 4, pp. 375–391, Feb. 2008, doi: 10.1007/S10209-007-0102-Z.

[12] "Silence Speaks Has Created AI-Powered Signing Avatars for the Deaf | WIRED." Accessed: Aug. 30, 2025. [Online]. Available: https://www.wired.com/story/silence-speaks-deaf-ai-signing/

[13] S. Sarker and M. M. Hoque, "An Intelligent System for Conversion of Bangla Sign Language into Speech," *2018 International Conference on Innovations in Science, Engineering and Technology, ICISET 2018*, pp. 513–518, Oct. 2018, doi: 10.1109/ICISET.2018.8745608.

[14] S. Das, M. S. Imtiaz, N. H. Neom, N. Siddique, and H. Wang, "A hybrid approach for Bangla sign language recognition using deep transfer learning model with random forest classifier," *Expert Syst Appl*, vol. 213, Mar. 2023, doi: 10.1016/J.ESWA.2022.118914.

[15] J. Uddin, F. N. Arko, N. Tabassum, T. R. Trisha, and F. Ahmed, "Bangla sign language

interpretation using bag of features and Support Vector Machine," *3rd International Conference on Electrical Information and Communication Technology, EICT 2017*, vol. 2018-January, pp. 1–4, Jul. 2017, doi: 10.1109/EICT.2017.8275173.

[16] N. Begum *et al.*, "Borno-Net: A Real-Time Bengali Sign-Character Detection and Sentence Generation System Using Quantized Yolov4-Tiny and LSTMs," *Applied Sciences 2023, Vol. 13, Page 5219*, vol. 13, no. 9, p. 5219, Apr. 2023, doi: 10.3390/APP13095219.

[17] M. J. ; Raihan *et al.*, "Bengali-Sign: A Machine Learning-Based Bengali Sign Language Interpretation for Deaf and Non-Verbal People," *Sensors 2024, Vol. 24, Page 5351*, vol. 24, no. 16, p. 5351, Aug. 2024, doi: 10.3390/S24165351.

[18] S. Siddique, S. Islam, E. E. Neon, T. Sabbir, I. T. Naheen, and R. Khan, "Deep Learning-based Bangla Sign Language Detection with an Edge Device," *Intelligent Systems with Applications*, vol. 18, May 2023, doi: 10.1016/J.ISWA.2023.200224.

[19] M. Hadiuzzaman, M. S. Ali, T. Sultana, A. R. Shafi, A. S. M. Miah, and J. Shin, "BAUST Lipi: A BdSL Dataset with Deep Learning Based Bangla Sign Language Recognition," pp. 280–285, Aug. 2024, doi: 10.1145/3723178.3723215.

[20] H. A. Rubaiyeat, H. Mahmud, A. Habib, and Md. K. Hasan, "BdSLW60: A Word-Level Bangla Sign Language Dataset," Feb. 2024, Accessed: Aug. 30, 2025. [Online]. Available: https://arxiv.org/pdf/2402.08635

[21] N. Haque, M. Serker, and T. Bin Bashar, "BdSpell: A YOLO-based Real-time Finger Spelling System for Bangla Sign Language," Sep. 2023, Accessed: Aug. 30, 2025. [Online]. Available: https://arxiv.org/pdf/2309.13676

[22] J. Ahmed, B. Shawon, H. Mahmud, and K. Hasan, "Fine-Tuning Video Transformers for Word-Level Bangla Sign Language: A Comparative Analysis for Classification Tasks," Jun. 2025, Accessed: Aug. 30, 2025. [Online]. Available: https://arxiv.org/pdf/2506.04367

[23] K. H. Tarafder, N. Akhtar, M. M. Zaman, M. A. Rasel, M. R. Bhuiyan, and P. G. Datta, "Disabling hearing impairment in the Bangladeshi population," *Journal of Laryngology and Otology*, vol. 129, no. 2, pp. 126–135, Feb. 2015, doi: 10.1017/S002221511400348X.

[24] "(PDF) Challenges of Educating Students who are Deaf and Hard-of-Hearing in Jordan." Accessed: Aug. 30, 2025. [Online]. Available: https://www.researchgate.net/publication/236866061_Challenges_of_Educating_Students_who_are_Deaf_and_Hard-of-Hearing_in_Jordan

[25] K. H. Tarafder, N. Akhtar, M. M. Zaman, M. A. Rasel, M. R. Bhuiyan, and P. G. Datta, "Disabling hearing impairment in the Bangladeshi population," *Journal of Laryngology and Otology*, vol. 129, no. 2, pp. 126–135, Feb. 2015, doi: 10.1017/S002221511400348X.

[26] K. H. Tarafder, N. Akhtar, M. M. Zaman, M. A. Rasel, M. R. Bhuiyan, and P. G. Datta, "Disabling hearing impairment in the Bangladeshi population," *J Laryngol Otol*, vol. 129, no. 2, pp. 126–135, Feb. 2015, doi: 10.1017/S002221511400348X.

[27] S. Tan, T. Miyazaki, and K. Nakadai, "Multilingual Gloss-free Sign Language Translation: Towards Building a Sign Language Foundation Model," May 2025, Accessed: Aug. 31, 2025. [Online]. Available: https://arxiv.org/pdf/2505.24355

[28] E. Fish and R. Bowden, "Geo-Sign: Hyperbolic Contrastive Regularisation for Geometrically Aware Sign Language Translation," May 2025, Accessed: Aug. 31, 2025. [Online]. Available: https://arxiv.org/pdf/2506.00129

[29] S. Fang *et al.*, "SignDiff: Diffusion Model for American Sign Language Production," Aug. 2023, Accessed: Aug. 31, 2025. [Online]. Available: https://arxiv.org/pdf/2308.16082

[30] B. Saunders, N. C. Camgoz, and R. Bowden, “Everybody Sign Now: Translating Spoken Language to Photo Realistic Sign Language Video,” Nov. 2020, Accessed: Aug. 31, 2025. [Online]. Available: https://arxiv.org/pdf/2011.09846

[31] A. S. Dhanjal and W. Singh, “Multilingual speech to Indian sign language translation using synthetic animation: a resource-efficient approach,” *Multimed Tools Appl*, vol. 84, no. 21, pp. 24637–24669, Jun. 2025, doi: 10.1007/S11042-024-20520-4.

[32] R. H. Pranto and S. Siddique, “Real-time Bangla Sign Language Translator,” *2024 27th International Conference on Computer and Information Technology, ICCIT 2024 - Proceedings*, pp. 2494–2499, Dec. 2024, doi: 10.1109/ICCIT64611.2024.11021821.

[33] “(PDF) A Rule Based System for Bangla Voice and Text to Bangla Sign Language Interpretation.” Accessed: Aug. 30, 2025. [Online]. Available: https://www.researchgate.net/publication/346037867_A_Rule_Based_System_for_Bangla_Voice_and_Text_to_Bangla_Sign_Language_Interpretation?utm_source=chatgpt.com

[34] G. Ferraro, R. Nazar, M. Alonso Ramos, and L. Wanner, “Towards advanced collocation error correction in Spanish learner corpora,” *Lang Resour Eval*, vol. 48, no. 1, pp. 45–64, Jul. 2014, doi: 10.1007/S10579-013-9242-3/FIGURES/3.

[35] “IEEE Transactions on Applied Superconductivity information for authors,” *IEEE Transactions on Applied Superconductivity*, vol. 28, no. 7, pp. C4–C4, Sep. 2018, doi: 10.1109/TASC.2018.2848159.

[36] A. H. Aliwy and A. A. Alethary, “Development of arabic sign language dictionary using 3D avatar technologies,” *Indonesian Journal of Electrical Engineering and Computer Science*, vol. 21, no. 1, pp. 609–616, Jan. 2021, doi: 10.11591/IJEECS.V21.I1.PP609-616.

[37] S. Siddique, S. Islam, E. E. Neon, T. Sabbir, I. T. Naheen, and R. Khan, “Deep Learning-based Bangla Sign Language Detection with an Edge Device,” *Intelligent Systems with Applications*, vol. 18, p. 200224, May 2023, doi: 10.1016/J.ISWA.2023.200224.

[38] Md. Sanzidul Islam, S. Sultana Sharmin Mousumi, N. A. Jessan, A. Shahariar Azad Rabby, and S. Akhter Hossain, “Ishara-Lipi: The First Complete MultipurposeOpen Access Dataset of Isolated Characters for Bangla Sign Language,” in *2018 International Conference on Bangla Speech and Language Processing (ICBSLP)*, IEEE, Sep. 2018, pp. 1–4. doi: 10.1109/ICBSLP.2018.8554466.

[39] S. Siddique, S. Islam, E. E. Neon, T. Sabbir, I. T. Naheen, and R. Khan, “Deep Learning-based Bangla Sign Language Detection with an Edge Device,” *Intelligent Systems with Applications*, vol. 18, p. 200224, May 2023, doi: 10.1016/j.iswa.2023.200224.

[40] S. Das, Md. S. Imtiaz, N. H. Neom, N. Siddique, and H. Wang, “A hybrid approach for Bangla sign language recognition using deep transfer learning model with random forest classifier,” *Expert Syst Appl*, vol. 213, p. 118914, Mar. 2023, doi: 10.1016/j.eswa.2022.118914.

[41] S. K. Akash, D. Chakraborty, M. M. Kaushik, B. S. Babu, and Md. S. R. Zishan, “Action Recognition Based Real-time Bangla Sign Language Detection and Sentence Formation,” in *2023 3rd International Conference on Robotics, Electrical and Signal Processing Techniques (ICREST)*, IEEE, Jan. 2023, pp. 311–315. doi: 10.1109/ICREST57604.2023.10070072.

[42] M. A. Rahaman, Md. H. Ali, and Md. Hasanuzzaman, “Real-time computer vision-based gestures recognition system for bangla sign language using multiple linguistic features analysis,” *Multimed Tools Appl*, Aug. 2023, doi: 10.1007/s11042-023-15583-8.

[43] M. S. Islam, A. J. M. A. Joha, M. N. Hossain, S. Abdullah, I. Elwarfalli, and M. M. Hasan, “Word level Bangla Sign Language Dataset for Continuous BSL Recognition,” Apr. 2023, Accessed: Aug. 30, 2025. [Online]. Available:

http://arxiv.org/abs/2302.11559
[44] T. Abedin, K. S. S. Prottoy, A. Moshruba, and S. Bin Hakim, “Bangla sign language recognition using concatenated BdSL network,” Jul. 2021, Accessed: Aug. 30, 2025. [Online]. Available: https://arxiv.org/pdf/2107.11818
[45] “(PDF) GESTURE RECOGNITION USING HIDDEN MARKOV MODEL.” Accessed: Aug. 30, 2025. [Online]. Available: https://www.researchgate.net/publication/344401589_GESTURE_RECOGNITION_USING_HIDDEN_MARKOV_MODEL
[46] “(PDF) Recognizing Bengali Sign Language Gestures for Digits in Real Time Using Convolutional Neural Network.” Accessed: Aug. 30, 2025. [Online]. Available: https://www.researchgate.net/publication/349028421_Recognizing_Bengali_Sign_Language_Gestures_for_Digits_in_Real_Time_Using_Convolutional_Neural_Network
[47] “(PDF) An Android Communication Platform between Hearing Impaired and General People.” Accessed: Aug. 30, 2025. [Online]. Available: https://www.researchgate.net/publication/346487602_An_Android_Communication_Platform_between_Hearing_Impaired_and_General_People
[48] “(PDF) A Real-Time Appearance-Based Bengali Alphabet And Numeral Signs Recognition System.” Accessed: Aug. 30, 2025. [Online]. Available: https://www.researchgate.net/publication/332797988_A_Real-Time_Appearance-Based_Bengali_Alphabet_And_Numeral_Signs_Recognition_System
[49] P. K. Chowdhury, K. U. Oyshe, M. A. Rahaman, T. Debnath, A. Rahman, and N. Kumar, “Computer vision-based hybrid efficient convolution for isolated dynamic sign language recognition,” *Neural Comput Appl*, vol. 36, no. 32, pp. 19951–19966, Nov. 2024, doi: 10.1007/S00521-024-10258-3.
[50] F. Yasir, P. W. C. Prasad, A. Alsadoon, A. Elchouemi, and S. Sreedharan, “Bangla Sign Language recognition using convolutional neural network,” *2017 International Conference on Intelligent Computing, Instrumentation and Control Technologies, ICICICT 2017*, vol. 2018-January, pp. 49–53, Jul. 2017, doi: 10.1109/ICICICT1.2017.8342533.
[51] O. Koller, O. Zargaran, H. Ney, and R. Bowden, “Deep Sign: Hybrid CNN-HMM for Continuous Sign Language Recognition,” *Proceedings of the British Machine Vision Conference 2016*, 2016, Accessed: Aug. 30, 2025. [Online]. Available: https://openresearch.surrey.ac.uk/esploro/outputs/conferencePresentation/Deep-Sign-Hybrid-CNN-HMM-for-Continuous/99511412402346
[52] “(PDF) Object Recognition using SVM-KNN based on Geometric Moment Invariant.” Accessed: Aug. 30, 2025. [Online]. Available: https://www.researchgate.net/publication/236154375_Object_Recognition_using_SVM-KNN_based_on_Geometric_Moment_Invariant
[53] “Speech-to-Text AI: speech recognition and transcription | Google Cloud.” Accessed: Oct. 06, 2024. [Online]. Available: https://cloud.google.com/speech-to-text/?utm_source=google&utm_medium=cpc&utm_campaign=japac-ROA-all-en-dr-BKWS-all-all-trial-PHR-dr-1605216&utm_content=text-ad-none-none-DEV_c-CRE_662846573741-ADGP_Hybrid+%7C+BKWS+-+BRO+%7C+Txt+-AI+%26+ML-Speech+to+Text-gcp+audio+transcription+software-main-KWID_43700077156208078-kwd-1720635580779&userloc_9069458-network_g&utm_term=KW_google%20cloud%20audio%20transcription%20software&gad_source=1&gclid=CjwKCAjwx4O4BhAnEiwA42SbVGl_jTzmTbl0KW64ZhmWQNb9Nbrc7IlTDBGIkVtcfglb_g_arn7aVBoC-BMQAvD_BwE&gclsrc=aw.ds
[54] “SiS-Builder.” Accessed: Oct. 06, 2024. [Online]. Available: https://sign.ilsp.gr/sisbuilder/demo.php

[55] A. Vaswani *et al.*, “Attention is All you Need,” *Adv Neural Inf Process Syst*, vol. 30, 2017.