Shahjalal University of Science and Technology
Department of Computer Science and Engineering

Bangla Text to Animated Sign Language Generation

MD. ASHIKUL ISLAM
Reg. No.: 2017331040
4^{th} year, 2^{nd} Semester

PRATO DEWAN
Reg. No.: 2017331106
4^{th} year, 2^{nd} Semester

Department of Computer Science and Engineering

Supervisor

DR. MOHAMMAD SHAHIDUR RAHMAN
Professor
Department of Computer Science and Engineering

4th March, 2023

Bangla Text to Animated Sign Language Generation

A Thesis submitted to the Department of Computer Science and Engineering, Shahjalal University of Science and Technology, in partial fulfillment of the requirements for the degree of Bachelor of Science in Computer Science and Engineering.

By

Md. Ashikul Islam
Reg. No.: 2017331040
4^{th} year, 2^{nd} Semester

Prato Dewan
Reg. No.: 2017331106
4^{th} year, 2^{nd} Semester

Department of Computer Science and Engineering

Supervisor

Dr. Mohammad Shahidur Rahman
Professor
Department of Computer Science and Engineering

4th March, 2023

Recommendation Letter from Thesis Supervisor

The thesis entitled *Bangla Text to Animated Sign Language Generation* submitted by the students

1. Md. Ashikul Islam
2. Prato Dewan

is under my supervision. I, hereby, agree that the thesis can be submitted for examination.

Signature of the Supervisor:

Name of the Supervisor: Dr. Mohammad Shahidur Rahman

Date: 4th March, 2023

Certificate of Acceptance of the Thesis

The thesis entitled *Bangla Text to Animated Sign Language Generation* submitted by the students

1. Md. Ashikul Islam
2. Prato Dewan

on 4th March, 2023, hereby, accepted as the partial fulfillment of the requirements for the award of their Bachelor Degrees.

Head of the Dept.	Chairman, Exam. Committee	Supervisor
Md Masum	Md Masum	Dr. Mohammad Shahidur
Professor	Professor	Rahman
Department of Computer	Department of Computer	Professor
Science and Engineering	Science and Engineering	Department of Computer
		Science and Engineering

Abstract

People with disabilities often face numerous challenges in their everyday lives. Despite the fact that everyone should have equal rights and opportunities in society, people with hearing impairments frequently experience social barriers and limitations. They typically use sign language and gestures to interact with others and convey their thoughts, which requires the presence of experts in this field who can accurately understand and interpret their language. Unfortunately the scarcity of interpreters has made it difficult for the hearing impaired community to communicate effectively and has created a significant gap in communication. In response to this challenge, the goal of this research work is to build such an automated system that can translate Bangla text into sign language in order to improve the social communication skill of the hearing impaired individuals. It is a web platform that can make it incredibly simple to teach and learn sign language. This system enables the generation of sign language from a given text, which can be displayed through an avatar. This system has the potential to improve social communication skills for individuals with hearing impairments, as well as for those without such impairments. By providing a means of conveying information in sign language, the platform can help to bridge the communication gap between hearing individuals and those with hearing impairments.

Keywords: Sign Language (SL), Bangla Sign Language (BDSL), Lemmatization, HamNoSys, SiGML, text to Sign Language, HamNoSys-based corpus

Acknowledgements

This report is prepared based on sign language generation from bangla text. We are very thankful to our honorable teachers and our department. We are also very grateful to our honorable supervisor Dr. Mohammad Shahidur Rahman, Professor, Department of Computer Science and Engineering, Shahjalal University of Science and Technology, Sylhet. He gave us really helpful guidance for our work and provided us with all the required resources. We are very pleased to work under his supervision. We are also very thankful to Hamburg University for providing us a helpful platform for generating required HamNoSys notations.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

The widespread adoption of information and communication technology (ICT) has transformed virtually all aspects of human life. It has revolutionized the way people work, conduct business, study, travel and communicate. In light of this, it is important to examine how ICT can be harnessed to effectively assist individuals with hearing loss in overcoming communication barriers. By leveraging the potential of ICT, we can work towards bridging the communication gap for the hearing impaired community and enhancing their overall quality of life. More than 5% of the total population of the world has hearing loss problem and requires help with their hearing difficulty [4]. The number is increasing day by day.

The primary means of communication for the hearing impairment people is sign language. They have their own grammatical rules and linguistic patterns, just like spoken languages. A type of interaction using hand gestures, facial expressions, motions, and body language is called sign language, commonly referred to as gestural language. Such gestures can be used to convey anything that can be said in spoken language. The Broca's and Wernicke's regions of the left hemisphere of the brain process sign languages similarly to other natural languages since sign language is not considered a global language [5].

Various sign languages including American, English, Indian, Arabic, Persian and others, are used around the world [6–8]. The own language of the hearing impaired people of our country is 'Bangla Sign Language' (BDSL). They deal with social issues like isolation since the hearing community doesn't provide them with enough communication support. To fulfill such connection gap interpreters are needed. Unfortunately, well-skilled interpreters are scarce for our country. To remove the obstacles, a translation system is essential to translate spoken language into sign language. Though some efforts

have been made in Bangla Sign Language generation, those systems are memory inefficient because of using pre-recorded videos [9]. Furthermore, those research works were basically based on a limited portion of words only, which are not enough to express the meaning of spoken sentences for any language.

In this regard, we are presenting a way of communication for the hearing impaired individuals in which a Bangla text sentence is translated into animated Bangla Sign Language. It is such an online web platform which can be used in various public services of our country (e.g. news channels, railway station etc). The potential benefits of this platform are particularly significant for hearing impaired individuals, who will be able to leverage its capabilities to generate sign language from text sources using an avatar.

1.1 Objectives

The main objective of our work is to develop a user friendly system that will allow users to learn Bangla Sign Language easliy. Users can enter a whole text into the system. The necessary sign language will then be generated in order to express the provided text. This system will be useful for those who find it challenging to communicate with those who have hearing loss and do not know the Bangla Sign Language. Our suggested system can be used for communication and information exchange in different places, including educational institutions, banks, bus and train terminals. Other objectives include -

- The creation of the first Bangla Sign Language generation system with a real-time 3D avatar.
- Developing the first Bangla Sign Language corpus containing almost 4000 signs.
- Sign language generation using the state of the art techniques.

Chapter 2

Background Study

2.1 Bangla Sign Language (BDSL)

Sign language is generally used for non-verbal communications. It is beneficial for hearing and speech impaired persons and for individuals with communication disorder. We must properly understand all the sign linguistic features and principles in order to utilize sign language in an effective and exact manner.

The components of bangla sign language can be divided into two categories:

- Characteristics
- Primary Elements

2.1.1 Characteristics

People use gestures, lip movements, and oral expressions while communicating with the hearing loss people. By using these non-manual parts of the body, sign language can be expressed more accurately.

2.1.1.1 Gestures

Different types of gestures can be used in any sign language, for example: Shaking the head back and forth or up and down conveys “yes” or “agreed,” whereas side-to-side shaking conveys “no” or

"negative". Moreover, a shrug of the shoulders and a lowering neck indicates the "carelessness" and "give up" respectively.

2.1.1.2 Lip Reading

Although when some words are said in oral language without being spoken, it may be realized simply by watching the movements of the lips. Understanding the movements of the lips without hearing the words is called "Lip reading". When expressing oneself through such gestures, consideration must be taken to avoid covering the face with anything.

2.1.1.3 Facial Expression

There are many changes in a person's face during communication, whether it is speech or sign language; some are obvious, while others are difficult to notice. During normal conversation, linguistic expressions vary based on the person's mood. These changes express the person's emotions, including happiness, grief, surprise, fear, and emotional expressiveness etc. Effective linguistic expressions in gestural communication includes, widening eyes, narrowing eyes, closing eyes, blinking eyes, frowning face, puffing cheeks, wagging tongue, rounding lips etc. Thus, a meaningful conversation is greatly impacted by the proper use of such facial expression.

2.1.2 Primary Elements

There are five fundamental elements of Bengali Sign Language. These are:

1. Hand Shape
2. Direction of Hand
3. Position of Palm
4. Movement of Hand
5. Position of Hand

To develop an effective communication, all expression-related components must be applied correctly. Any improper presentation of these elements will lead to a gesture that alters the intended meaning of what was spoken.

Usually each sign language has its own fundamental elements; for instance, American Sign Language (ASL) has three basic elements:

1. Tabula, usually known as “Tab”, stands for location.
2. Designator, which stands for Configuration, abbreviated as “Dez”.
3. Signation, which means Movement, abbreviated as “Sig”.

Another basic element is added in British Sign Language (BSL) which is called Orientation.

2.1.2.1 Hand Shape

The hand is the primary tool used in sign language. The major task is done with the use of one hand or two hands. There are more than 30 different hand shapes required for representing Bangla Sign Language. Each representation requires a different hand shape. Depending on the form, different names are given to hand shapes.

Figure 2.1: BDSL Hand Shapes [1]

2.1.2.2 Direction of Hand

Direction is another important element of sign language. The same shape may have different gestures for different hand directions. Some hand shapes need multiple fingers and their tip positions may vary. Depending on which finger is used, the location of that finger's tip will determine the direction of the hand.

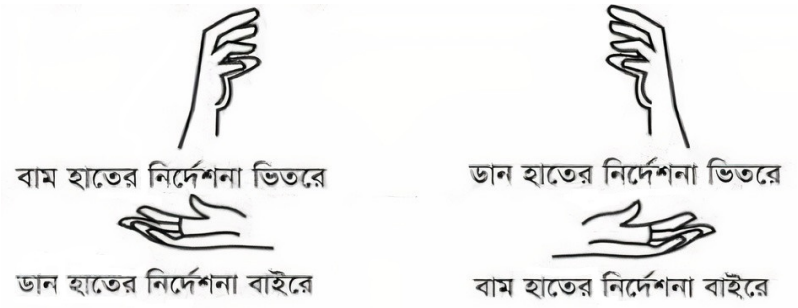

Figure 2.2: BDSL Hand Direction [1]

2.1.2.3 Position of Palm

Another major element of sign language is the palm. Variations in hand shape and direction result in different signs. Similar to this, different palm position also represents different signs.

Figure 2.3: BDSL Hand Palm [1]

2.1.2.4 Movement of hand

Sign language is a language that depends on hand shape, direction, palm, movement etc. Hand movements are very important in sign language. Hands may remain still in representing many gestures. Again, in many cases, the hands need to move in different directions one or more times.

2.1.2.5 Position of Hand

Hand position is also important for the correct expression of sign language. Hand position refers to the height at which the hand will be positioned, such as the waist, the shoulders, the chest and the forehead. The main reason for including height is because the same hand shape can represent different signs at different heights.

2.2 Bangla Root Words

With approximately 230 million native speakers, Bangla is one of the world's most commonly spoken languages. It is an Indo-Aryan language that is the state language of Bangladesh [10]. Bangla is one of the hardest languages to lemmatize due to high level of inflections. It is difficult to create many NLP(Natural Language Processing) systems due to the high degree of inflections, such as word sense disambiguation (WSD). It is challenging to implement conventional dictionary-based WSD techniques for any such inflected language. Many NLP systems have not yet been created for Bangla because this language lacks a proper lemmatizer. New opportunities for fields, such as automated language translation and grammatical correction to fourish in Bangla, will be opened up by a dependable Bangla lemmatizer.

An inflected word in Bangla can exist in various forms. For example, consider the following words: "সুন্দর" (sundor: Adjective(beautiful) / Noun(beautiful man)), "সুন্দরী" (sundori: Noun(beautiful woman)), "সুন্দরভাবে" (sundorvabey: Adverb(in a beautiful way)), "সৌন্দর্য" (soundorjo: Noun (beauty)). These terms are all variations of the word "সুন্দর".

Inflected words follow regular pattern in which a surface word is created by adding or subtracting frequent Case, Suffix, Inflexion, Number and Article from the base word [11]. For instance, 'ছাত্ররা'(chatro): 'রা'(ra) is added with 'ছাত্র'(chatro) to produce the infected word. Similarly, 'পেন্সিল'(pencil) from 'পেন্সিলটি'(pencilti), 'পেন্সিলটাকে'(penciltake) are some examples of infection and base form.

The following are some methods that a Bangla inflected word can be formed:

- By removing a valid suffix from the lemma (e.g., ‘পড়’ = ‘পড়া’(to read) - ‘া’, ‘চল’ = ‘চলা’(to walk) - ’া’).
- By adding a valid suffix with the lemma (e.g., ‘জাপানি’ = ‘জাপান’(Japan) + ‘ই’).
- By subtraction and followed by addition with the lemma(e.g., ‘পড়ছেন’ = [‘পড়া’ (to read) - ‘া’] + ‘ছেন’).
- By adding article with the lemma (e.g., ‘কলমটি’ = ‘কলম’(Kolom) + ‘টি’).
- By adding inflection with the lemma (e.g., ‘নিয়মের’ = ‘নিয়ম’(rules) + ‘এর’).

Another variety produces irregularly inflected terms for which there exists no effective translation formula.

- None of the above transformations works, (e.g., ‘যাওয়া(to go)’ -> ‘গিয়েছিলো (went)’).

A root word’s morphological variations are typically orthographically and semantically close to the root term. In this instance of irregular transmission, the only connection between a lemma and its variations is semantic. According to this rule, no Case, Suffix, Number, Article additions or subtractions are used to create the surface term.

Chapter 3

Related Works

3.1 Sign Language

The development of sign language generation systems has been a major focus of study for the last few decades. In this section, significant achievements in this field of research are highlighted.

Ozawa et al. [12], proposed a system that generates Japanese Sign Language from image and text processing. This system uses several multi-modal information processing system . In 3D virtual reality, animations are produced using avatar movement. To produce better animation, text is processed and sign language for each category is combined.

A computer-based system that uses sign language gesture parameters to produce VRML, or Virtual Reality Modeling Language animation sequences, was proposed by Papadogiorgaki et al. [13]. They have worked with 3200 words in their system. After processing a sentence, the output frame displays the gesture that was saved.

A system that uses Bangla text as input and displays sign language gestures as output was developed by Sarkar et al. [14]. A significant number of sign language gestures are recorded as a dataset for Bangla words and sentences. The processed input text is compared to the previously saved text in order to map the appropriate video gesture to the input text.

Porta et al. [15] created a translation system using rule-based approach for translation from Spanish text to Spanish sign language. The implementation of the transfer-based architecture was completed using syntactic functions of dependency analysis.

Hoque et al. [9] suggested a two-way communication system between Bangla and BDSL. The system

uses pre-recorded and some predefined rules for bangla text to bangla sign language generation.

A technique was created by Megha Vargese et al. [16] to produce sign language from English text. English text is first converted into Hamnosys notation. Then, the notation is converted to xml based SiGML. Finally SiGML notation produces animation using motions from an avatar.

A method for automated sign language was suggested by Tiago Oliveria et al. [8]. In this method, grammatical rules are applied to the input text. The rules categorize a phrase into order of word, verb and tense. Then compare the output with a database of sign language. The avatar performs animation after returning the gesture.

Stoll et al. [17] preseneted a Neural Machine Translator and generative Adversarial Networks as a method for producing sign language from spoken speech. NMT helps in the prediction of word sequences and the modeling of the full sentence into a single model. It produces animation using motion graph. However, this strategy cannot compete with current avatar systems.

Based on ISL grammar, Kumar et al. [6] created a real-time system for creating sign language. The presented system generates sign language using techniques like HamNoSys, SiGML, and avatar animation. Similarly, the same techniques have been used to produce other sign languages like persian sign language and pakistani sign language [7, 18].

There are also real-time sign language translation systems that can convert English text/words into American Sign Language (ASL) and vice versa [19]. The system also has the capability to detect sign languages that is performed by any user. To detect the signs, a sequential model (e.g. convolutional neural network model) was trained using a manually constructed ASL dataset. The system is also able to generate sign languages from speech data. Additionally, Joy and Balakrishnan proposed a prototype which is able to convert Malayalam languages to Indian Sign Language (ISL) from any Malayalam text [20]. Here, a POS (parts-of-speech) tagger has been used for linguistic analysis as well as sentence optimization.

3.2 Lemmatizer

Lemmatization is necessary for our approach to find the root words. On languages, such as English, Chinese, French, a lot of effort has been done. Although languages like Hindi and Bangla are widely spoken, they have more complex internal structures and a wider range of morphological word forms than other languages. Little research has been done on these languages in the Natural Language Processing field because of their structural complexity and extensive morphological variation. Some of the established methodologies are described briefly.

3.2.1 Foreign Lemmatizers

In 2013, Snigdha Paul et al. [21] created a lemmatizer for the Hindi language. They have suggested a rule-based method for extracting the suffixes from a word in order to find the lemma. They manually finished their task and developed 112 rules using a dataset of roughly 20,000 sentences.

Zeman et al. [22] suggested creating a collection of rules that convert the surface form into a lemma in order to create a lemma. These guidelines may include extra guidelines for modifying, duplicating, moving, or removing a character from the surface form. The lemmatization task is transformed into a multi-class classification task as a result, and the generated rules of training set are used to make a collection of all possible transformation rules from which the correct one is chosen.

A system called Nefnir was suggested by Svanhvit Ingolfsdottir et al. [23], which they have compared with two other lemmatizers,a rule-based lemmatizer named CST (Center for SprogTeknologi/Center for Language and Technology) [24] and a part of IceNLP toolkit named Lemmald [25] . It mainly applies the suffix substitutions rule. It lemmatizes tagged texts, and for labeling, they used both manually written POS tags and an automated POS tagger.

A random forest classification model for language independent lemmatization was presented by Akhmetov et al. [26]. The writers of this article suggest building character cooccurrence embeddings from pairs of infected words-lemma from 25 different languages. The decision tree method is then used with this character co-occurrence encoding to produce the final model.

For some chosen languages, a lemmatizer that combines a dictionary-based lemmatizer and a seq2seq-based [27] lemmatizer is built as part of the Stanza toolset [28]. An attention layer with soft dots, a greedy decoder, and a bidirectional LSTM encoder are used to build this seq2seq-based lemmatizer. Milintsevich et al. [29] make an effort to better the Stanza lemmatizer's performance by utilizing an

external system that provides additional lemma options, but they only slightly outperform the original system for the 23 supported languages.

3.2.2 Bangla Lemmatizers

A rule-based lemmatizer, the Bengali morphological analyzer developed by Faridee et al. [30] can only lemmatize terms that are commonly used in the language. Additionally, the word lemmas generated by their technique cannot be connected to standard dictionaries because their method is basically stemming.

For Bangla nouns, Alok Ranjan Pal et al. [31] proposed lemmatization method. In order to find the root word, they utilized a stripping technique in which the longest suffixes were reduced. The system processed the input nouns after they had been classified in a lexicon to produce the base form.

For the Bengali language, Akshay Chaturvedi et al. [32] have suggested a neural lemmatizer. They have produced 200-dimensional word vectors produced using the word2vec tool. Their proposed lemmatization model is language independent.

Benlem is another lemmatization method developed by Abhishek Chakrabarty et al. [11]. Benlem is a rule-based lemmatizer for Bangla that improves accuracy by using POS-annotated infected words and a valid suffix set. It achieved 83.82% accuracy.

Bengali information retrieval system (BIRS) is an information retrieval system that was unveiled in 2019 by Md. Kowsher et al. [33]. In order to lemmatize Bangla verbs that cannot be lemmatized using rule-based algorithms, they have presented two innovative strategies. Dictionary Based Search by Eliminating Afx was the first strategy. The second was Trie, a data structure built on trees that can retrieve any lemma.

The approach Alok Ranjan Pal et al. [34] suggested for Bangla Word Sense Disambiguation is effective (WSD). They have used both supervised and unsupervised techniques throughout the process. Their procedure was more accurate in terms of lemmatization. However, they did not disclose their process for lemmatizing Bangla words.

Islam et al. [35] provided an efficient encoder-decoder-based sequence-to-sequence framework for deriving the lemma from an infected Bangla word while considering parts-of-speech as context. The suggested framework significantly outperformed other Bangla lemmatization methods with character accuracy of 95.75% and precise match of 91.81% on the testing split of the produced dataset.

Chapter 4

Methodology

The suggested method will use the BDSL grammar shown in Figure 4.1 to produce BDSL sentences from Bangla text. First Bangla words or sentences will be taken as input. Then the parser will be used to parse the input. As a result, root words with the corresponding morphological information will be produced. In this case, The BDSL rules will be applied to each of the input sentences. These BDSL grammar rules will all be used to construct an output and extract the word order. The appropriate SiGML for each retrieved root word will be obtained. After that, the animation server will receive it. The SiGML file's tags will determine the animation frames that are produced, which will then be put together for creating 3D avatar animation.

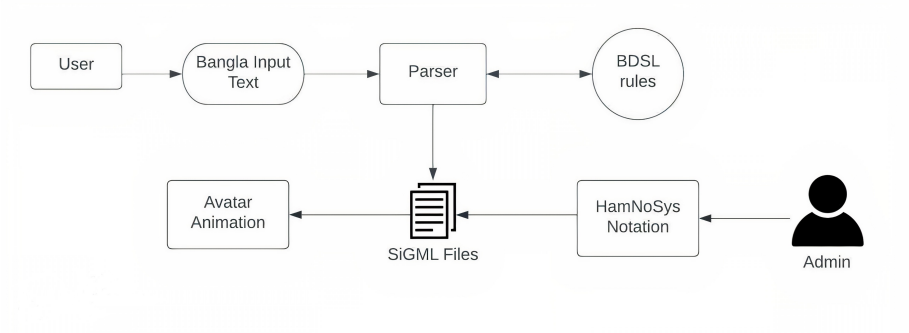

Figure 4.1: System architecture of our proposed system

4.1 HamNoSys Notation

In 1984, Siegmund Prillwitz developed the HamNoSys notation.It is a standard transcription system for expressing sign language (SL) that has nearly 200 characters and can be used anywhere [3]. It has both non-manual and manual parts of the body. An example of a phonetic notation is HamNoSys, which comprises of various factors, including the arrangement of handshape, location, movement, facial expressions and body gestures. Among other things, the factors are referred to as hand form, orientation, position and movement.

For a single sign, a HamNoSys notation includes a description of the starting posture containing

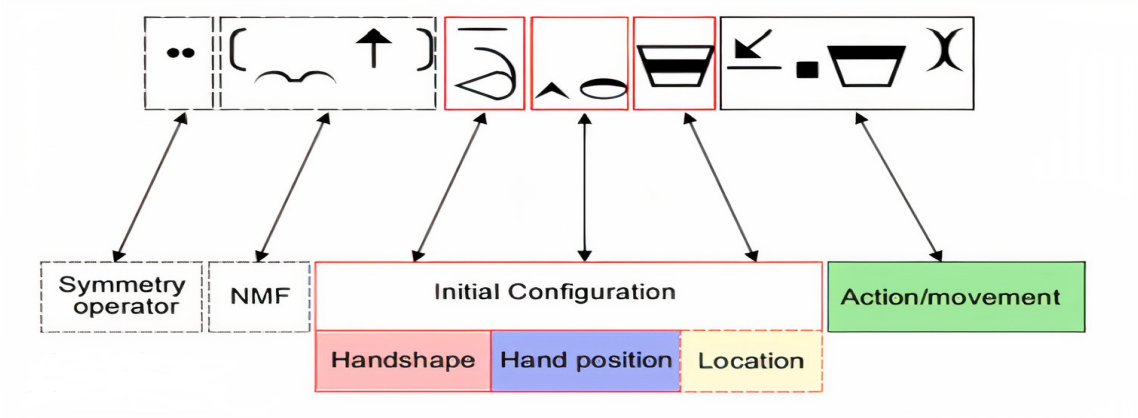

Figure 4.2: HamNoSys General Structure [2]

nonmanual attributes, handshape, hand direction, and hand position as well as the actions that change this posture sequentially. But for representing two handed signs, symmetry and non-symmetry operators are used for describing if the dominant and non-dominant hand should do things together or not.

4.1.1 Handshape

Diacritical marks for the position of the thumb and bending are included in the description of a handshape. Using symbols, changes from a specific structure involving the fingers or the hand shapes are also possible.

Basic handshapes and their correponding HamNoSys notation:

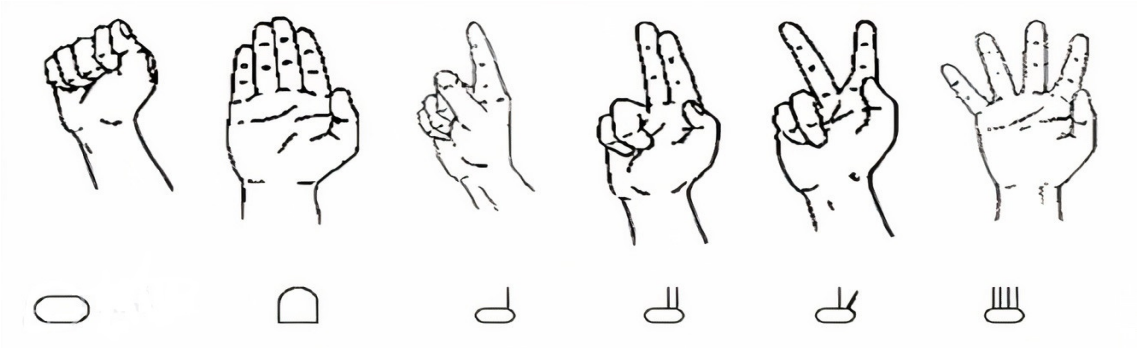

Figure 4.3: Basic Handshape Notations [2]

The above notations can be combined with diacritic signs for thumb position:

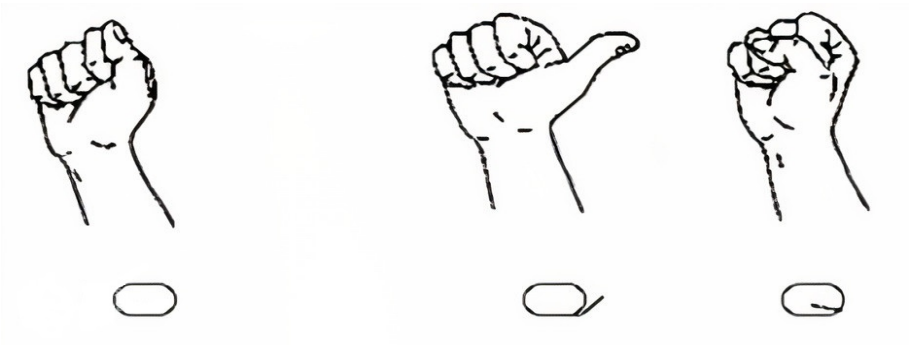

Figure 4.4: Basic Thumb Notations [2]

These notations can be used for bending the hand:

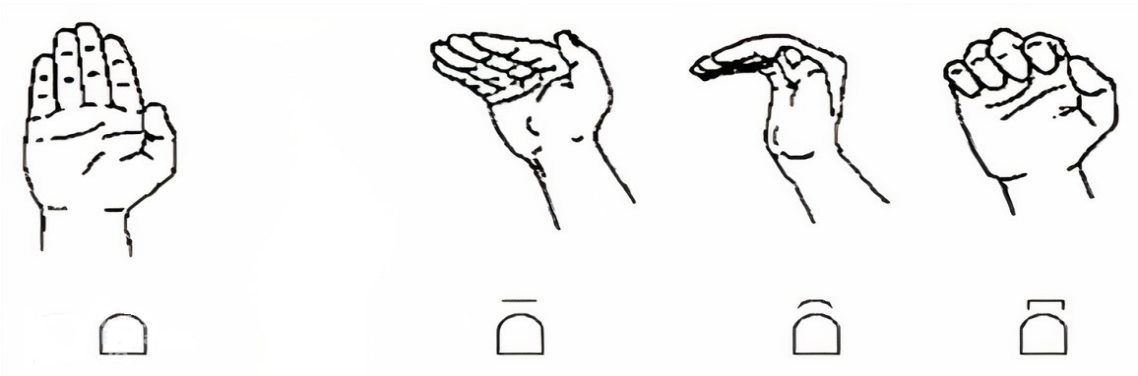

Figure 4.5: Basic Hand-bending Notations [2]

4.1.2 Orientation

Orientation defines the direction of finger and palm of a handshape. It is split up into 2 orientations: Finger base Orientation and Palm Orientation.

Finger base orientation:

The direction the fingers when completely extended corresponds to the orientation of the fingers.

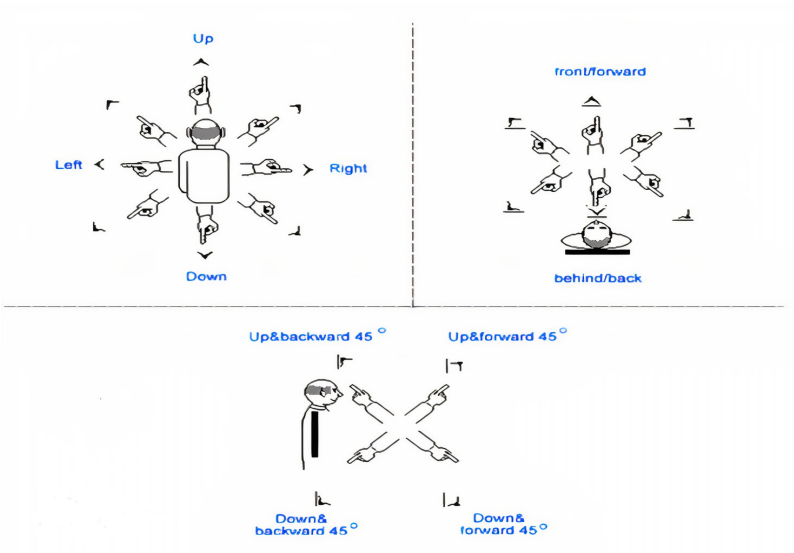

Figure 4.6: Hand Finger Orientation [2]

Palm orientation:

There are eight possible values for the palm's orientation, each of which indicates how the palm should be placed in relation to the hand's shaft.

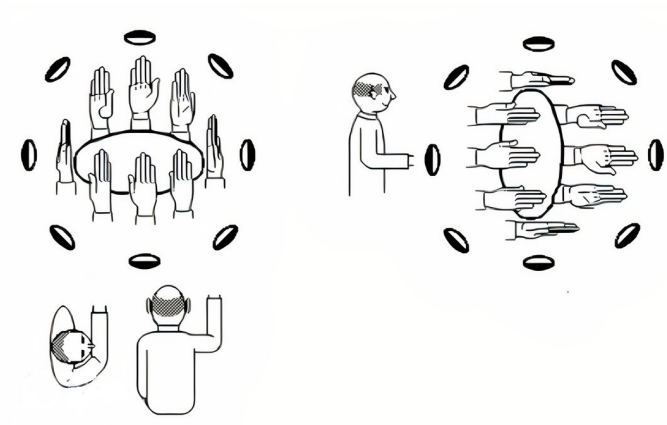

Figure 4.7: Hand Palm Orientation [2]

4.1.3 Location

The hand's placement on the body is indicated by the location symbols. A hand location can be specified with more than 40 symbols. Description of all possible location combinations:

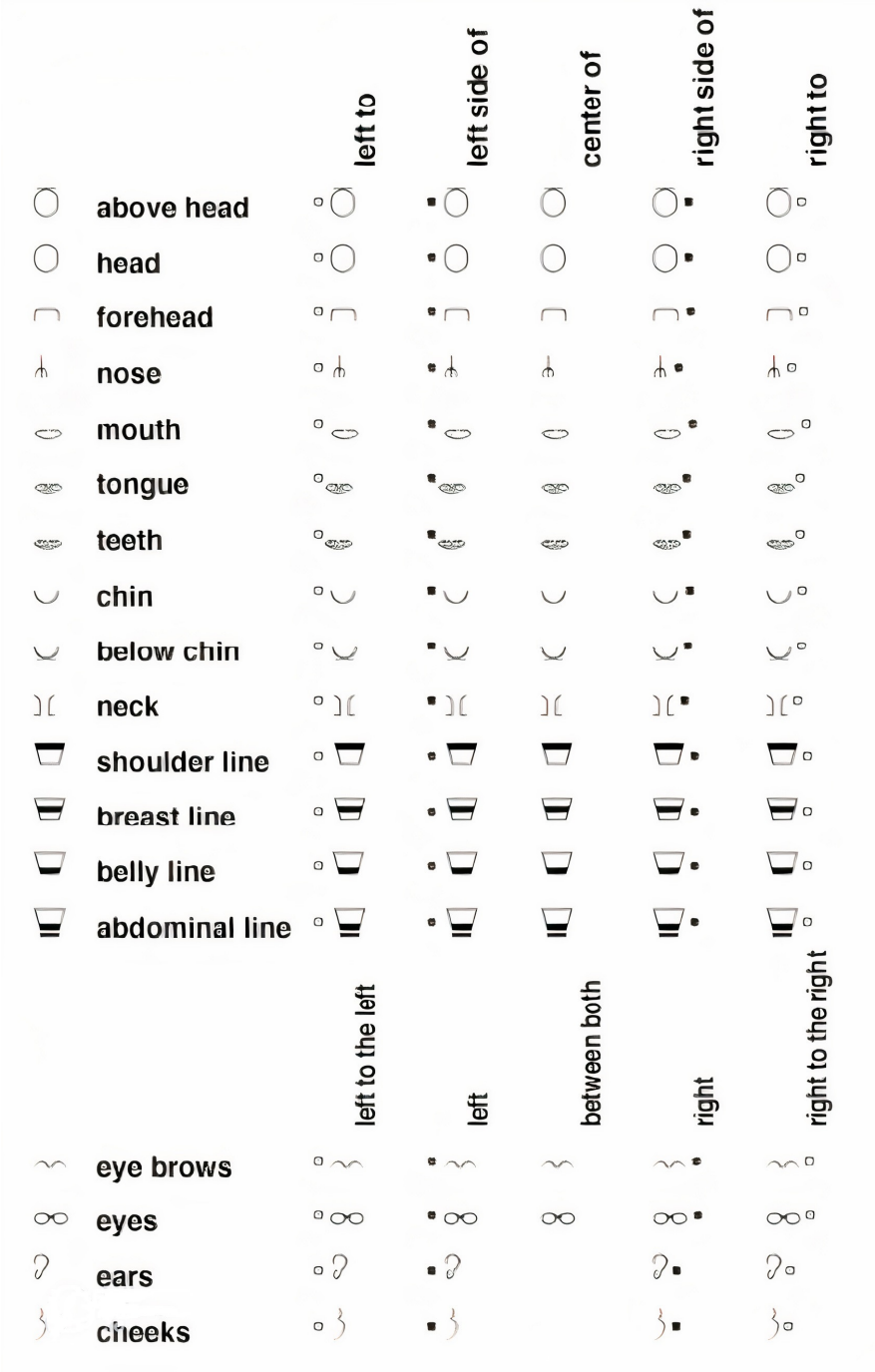

Figure 4.8: Location symbols & their meaning [3]

4.1.4 Movement

This sections indicates how many types of movements can be possible for the hands to represent any sign using these HamNoSys symbols. There are two types of movements.

- Straight movements

- Circular movements

Straight movements:

The description of movements in a straight line corresponds to the same rules as the description of the direction of the base of the forefinger. There are more than 20 symbols that are available for straight movements of the hand. Curved movement symbols are also included here.

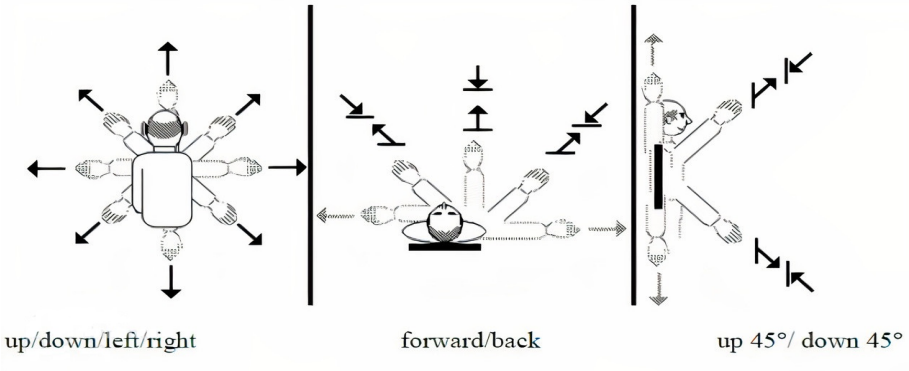

Figure 4.9: Straight Movement [2]

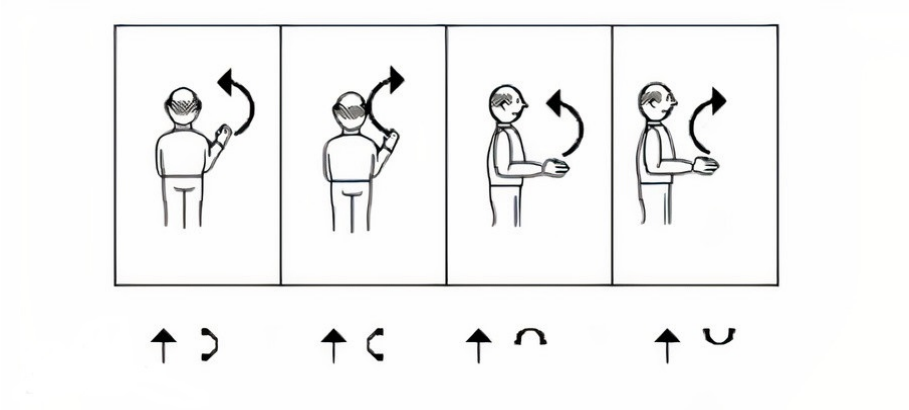

Figure 4.10: Curved Movement [2]

Circular movements:

HamNoSys provides three types representation for circular movements: horizontal-vertical in the plane parallel to the person, vertical in the plane and perpendicular to the person.

Circular movements indicating the directions with arrows:

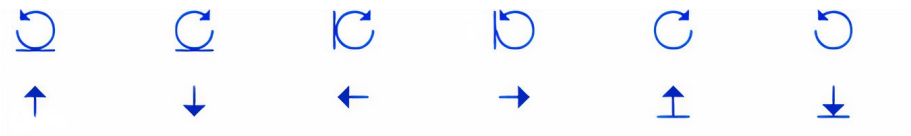

Figure 4.11: Circular Movement with respect to the direction [2]

Diagonal movements indicating the direction with arrows:

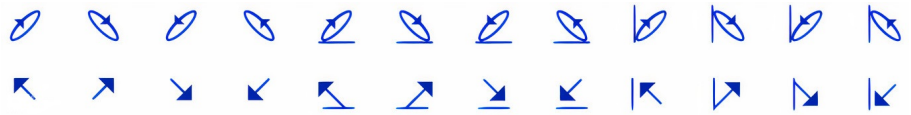

Figure 4.12: Diagonal Movement with respect to the direction [2]

4.1.5 Two-handed

Two-handed symbols are called symbols of symmetry. These symbols are provided only for two-handed signs. One hand is dominant and the other is non-dominant. The symbols for these signs must be included in the beginning. Symmetry symbols for two-handed signs:

Figure 4.13: Symmetry Operators [2]

Structure for different shape,location or orientation of both hands:

Figure 4.14: Non-Symmetry structure [2]

4.2 Sign Language Dataset

We now give a brief description of the dataset. The dataset is provided by the Committee on Standardization of Bangla Language for Use in Information Technology, which is headed by the Executive Director of the Bangladesh Computer Council (BCC), a statutory body of the ICT Division of the Ministry of Posts, Telecommunications, and IT. There are around four thousand signs in this dataset that can be used for non-verbal communications. The signs are distributed according to various categories. Furthermore, mobile applications are also available for these signs along with their corresponding pre-recorded videos.

4.3 System Overview

Basically our system is designed with four major components:

- Parser
- HamNoSys Notation Creation
- HamNoSys to SiGML file generation
- Animation Creation

4.4 Parser

A parser is a such a mechanism that is used for analyzing grammatical structure of a sentence in any language and generating the corresponding output. In the context of processing a Bangla sentence, the input text must be tokenized by the parser into individual words and then analyzes each word's morphology, syntax, meaning to determine the actual context of the sentence. Since our language is morphologically very rich with various dialects, the role of the parser in our system is much more challenging. However, the role of the parser in our system is:

- Removing unnecessary words from input text.
- Finding root words (Lemmas).

4.4.1 Removing Unnecessary Words

All the words present in a Bangla sentence have no separate meaning. There is also no sign language for such words as well. These unnecessary words might cause additional processing overhead while creating Bangla Sign Language from any Bangla input text. Conjunctions, prepositions and other filler words are such type of words that do not have their own meaning in our Bangla Sign Language. By removing unnecessary words from the input text, processing time may be significantly reduced, making our system more efficient. Furthermore, removing unnecessary words from the input can increase the accuracy and clarity of the final sign language by focusing on the most significant and meaningful words. Thus, we have excluded some manually enlisted unnecessary words from our system while processing the Bangla text.

4.4.2 Lemmatization

In order to use a given bangla text as input for our system, we must first identify all of the underlying lemmas. Such process involves identifying the underlying morphological meaning of a word, by reducing it to its base form. For example, consider the following bangla sentence: "বিকালের(Bikaler) আলোয়(Aloy) চারিপাশ(Charipash) আলোকিত(Alokito)". The expected lemmas for the sentence is:

বিকাল(Bikal); আলো(Alo); চারপাশ(Charpash); আলোকিত(Alokito).

To find out all the lemmas, we have used a sequence-to-sequence (seq2seq) RNN encoder-decoder model, which is extensively used for sequential data applications including machine translation, speech recognition, and text analysis. The architecture of this model is made up of two primary parts: an encoder and a decoder. The encoder accepts a sequence of inputs, such as a phrase in one language, and outputs a fixed-length vector known as the context vector, which summarizes the information in the input sequence. Based on the context vector and prior outputs, the decoder then takes the context vector and creates a series of corresponding lemmatized output. To train our model, we have used publicly available BaNeL dataset [35]. There are three column in the dataset. The first column contains inflected terms that show how the root word transforms depending on its grammatical variety. The second is the lemma, which indicates the base form for any inflected word. The final one is parts-of-speech. Hence, the dataset contains more than 22300 inflected words and 6300 root words (lemmas) with their associated parts-of-speech (POS) tags.

The following part goes over dataset preparation, a brief discussion of our model design, and the implementation process, as well as data manipulation using some examples. Furthermore, we developed a lemmatizer based on mapping and rule-based approach for comparison purposes.

4.4.2.1 Data preprocessing

As we know, several steps of encoding and preprocessing are required to make the data compatible with neural networks. We trained our model using character level encoding as a suitable data format. Such type of encoding retains the structural information of words, allowing the model to properly predict new unseen lemma if the type of inflection is known. For example: if the model has been trained with such inflected words: "করেছিল (korechilo)", "খেয়েছিল (kheyechilo)", "গিয়েছিলো (giyechilo)", then it can correctly predict the lemma of the similar type of word "পড়েছিল(porechilo)", which is "পড়া(pora)".

i) Character Mapping:

The Bangla alphabet consists of 50 letters. Of which, there are 11 vowels and 39 consonants. Hence, a complete Bangla alphabet list, as well as padding token as an ending symbol for any word, has been prepared. Each of these symbols is mapped with a unique integer shown in table 4.1.

ID	0	1	2	3	4	5	6	7	8	9	10
Character	অ	আ	ই	ঈ	উ	ঊ	ঋ	এ	ঐ	ও	ঔ
ID	11	12	13	14	15	16	17	18	19	20	21
Character	ক	খ	গ	ঘ	ঙ	চ	ছ	জ	ঝ	ঞ	ট
ID	22	23	24	25	26	27	28	29	30	31	32
Character	ঠ	ড	ঢ	ণ	ত	থ	দ	ধ	ন	প	ফ
ID	33	34	35	36	37	38	39	40	41	42	43
Character	ব	ভ	ম	য	র	ল	শ	ষ	স	হ	ড়
ID	44	45	46	47	48	49	50	51	52	53	54
Character	ঢ়	য়	ৎ	ঁ	ু	ৃ	ো	ৈ	ী	ঃ	ং
ID	55	56	57	58	59	60	61				
Character	া	ি	ু	ৃ	ে	ৌ	$				

Table 4.1: Character mapping set.

Thus a character mapping set with an indexing number of (0 to 61) has been created, where the 61th symbol ('$') represents the terminate symbol while processing the data.

ii) Integer encoding:

Following the preparation of the character mapping set, all inflected and root words in the dataset are divided into characters. Then, using the character index number from the list, those characters are converted to integer numbers, e.g. the word "অগভীরতা(ogovhirota)" is split into individual characters: ['অ','গ','ভ','ী','র','ত',' া']. Now, the integer encoding of these characters will be: [0, 13, 34, 52, 37, 26, 55]. Thus integer encoding has been created for all inflected and root words.

iii) One hot encoding with Masking:

For both inflected and root words in the dataset, we have converted the integer encoding of them into one-hot encoded format. We assumed that the maximum length of a word is 17 (*max_len*), which was later utilized in our model during training. As a result, the length of each integer encoding has been converted to *max_len*. For example: the integer encoded format of the word "অগভীরতা(ogovhirota)" is [0, 13, 34, 52, 37, 26, 55]. After padding this list upto *max_len*, the list will be [0, 13, 34, 52, 37, 26, 55, 61, 61, 61, 61, 61, 61, 61, 61, 61, 61]. The padded characters are represented by the 61th character (e.g. <$>) in our unique character mapping set. In fact, each of these integer digits are represented as 62 length vector. Thus, the dimension for each of the inflected and root word is: (*max_len* * (*unique_Bangla_character*)) = (17*62). Hence, the one-hot encoding for the word "অগভীরতা" = ['অ','গ','ভ','ী','র','ত',' া'] = [0, 13, 34, 52, 37, 26, 55] = [0, 13, 34, 52, 37, 26, 55, 61, 61, 61, 61, 61, 61, 61, 61, 61, 61] = [[1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],......,[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1]].

4.4.2.2 Proposed Model Architecture

After exploring the morphological diversity of bangla language, we have been motivated to use the "Sequence to Sequence Encoder Decoder" approach to design our lemmatization model. The encoder receives one hot encoded character stream as sequential input. The outputs of the encoder are then used by the decoder to generate a time-step aligned context vector, which is finally used to predict lemma characters sequentially.

a) Encoder:

Our encoder is a unidirectional multi-layer stacked Recurrent Neural Network (RNN) that receives a batch of one-hot encoded character inflected word streams as input at each time-step of the RNN. Here, we have shown our unidirectional Encoder model for a single layer in figure 2, where each of the RNN unit is LSTM (Long Short Term Memory). Each vector X_i represents the inflected word's one-hot encoded character streams. Then, these vectors X_i are sequentially given to each of the LSTM layer. As a result, there will be three outputs for each time step: two hidden states (exactly the same values) and one cell state:

- One hidden state's output can be used for prediction or to integrate another layer of LSTM.
- And the other hidden state and cell state has been used for the next time-step.

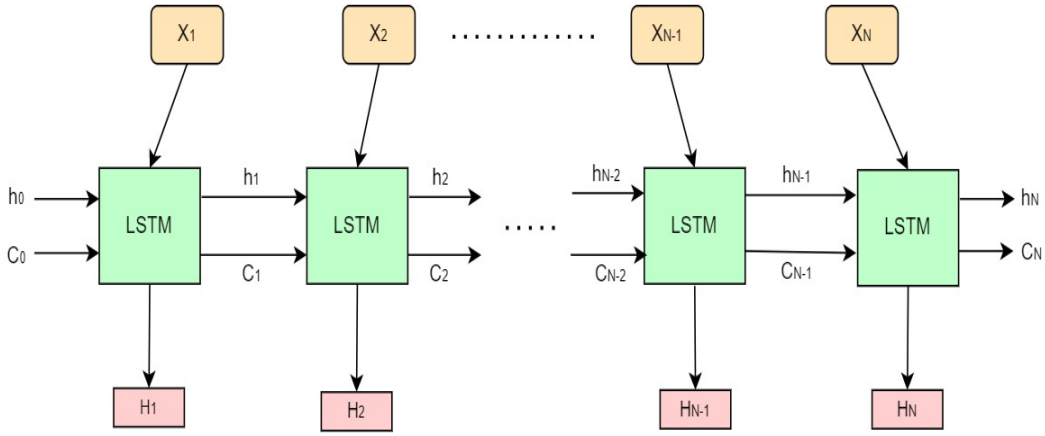

Figure 4.15: Encoder

According to figure 4.15, N indicates the length of the longest inflected word in the current batch. The encoder generates the context vectors after processing the encoded formatted input data. In fact, these context vectors are the concatenation of the last time step's hidden state and the last time step's cell state.

b) Decoder: The last cell's hidden state and cell state of the encoder will be processed as the initial state's context vectors for the decoder. In the decoding phase, a special symbol (e.g. 'START') is required to indicate the initial input. In a word, decoder will execute in a such way by utilizing its state and the given proper output as the next step context vector and input, until the generated output is a special symbol 'STOP' or the pre-defined maximum steps (length of output) is achieved. Furthermore, it utilizes the previous hidden state as the next input and the last cell states as the next states for itself. Lastly, the last hidden state of the current time step has been utilized to predict our desired lemmas. During the training phase, we trained our model applying teacher-force-ratio (TRF), which is a strategy

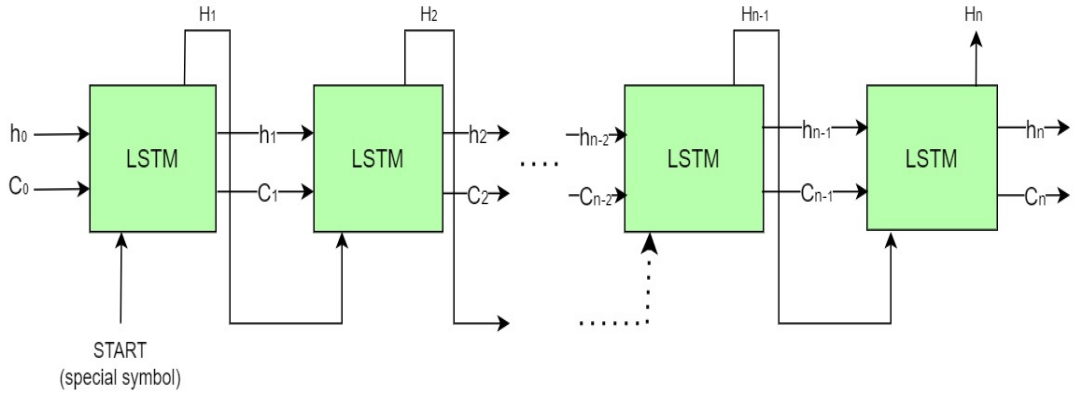

Figure 4.16: Decoder

for optimizing the training process by randomly skipping preceding decoder output and feeding the real target data as decoder input. Moreover, an additional dense layer with softmax activation function is used to predict the next output. This dense layer will output the one-hot encoded format data. Although our custom designed lemmatizer model can accurately predict lemmas for small words (about 78 characters), it is unable to predict lemmas correctly for large words (above 12 characters) and for proper nouns as well.

4.4.2.3 Word Mapping and Rule Based Approach

To identify the root words in the given Bengali text using mapping and rule-based lemmatization, the following steps are followed:

1. First, we check if the root form of each word in the input text is already present in the Banel dataset, which contains information about Bengali root words.
2. If the root word is found in the Banel dataset, we store the corresponding root word in an array.
3. If the root word is not found in the dataset, we check whether the word ends with a suffix (e.g. ‘ছিলাম’, ‘ছিল’, ‘ছিলে’, ‘টা’, ‘টি’, ‘কে’, ‘র’) or not. If it does, we remove the suffix.
4. After removing the suffix, we check again if the root form of that word is present in the dataset.
5. If the root word is still not found in the dataset, we return the resulting word as a root word.
6. Finally, we add the resulting root word to the array of root words for animation.

Therefore, to summarize, the process involves checking the Banel dataset for root words, removing suffixes and returning the root word. This method enables us to identify the root words in the given Bangla text.

4.5 HamNoSys Notation Creation

To animate any word, first of all we need to generate the appropiate hamnosys notation for that word. For this, we are using a open accessible software named as “HamNoSys” [36]. Different types of parameters can be easily configured in this software, including hand, movement, position, head, and body movements. These factors include hand form, hand orientation, position, and movement.
The hamnosys notation of the character আ:

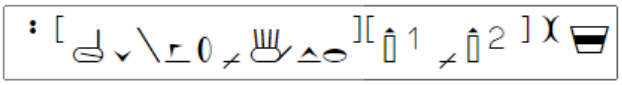

Figure 4.17: HamNoSys Notation of the character আ

The hamnosys notation of bangla character ‘আ’ is depicted in figure 4.17. The notaton indicates to produce such an animated sign where both hands will be used. It means keeping both hands at chest height, the thumb of the right hand is in contact with the index finger of the free left hand.
The hamnosys notation of the digit ০ (শূন্য):

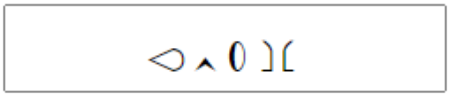

Figure 4.18: HamNoSys Notation of the digit ০ (শূন্য)

Here in figure 4.18 the closed pinch right hand is located at the neck position and the fingers pointing towards the left side of the body posture, thus indicating the sign for the digit ০ (শূন্য). The hamnosys notation of the word ‘বই’:

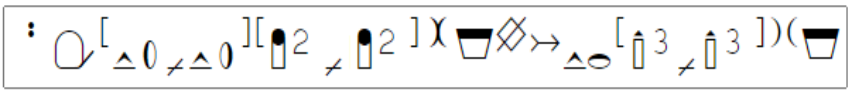

Figure 4.19: HamNoSys Notation of the word বই

To express the sign of the word ‘বই’, here we have to use the symmetry operator as it indicates the necessity of the both hands. Initially the orientation of both hands are exactly same, but in opposite of each other. And the index finger of both hands are touching each other as they are clasped together across the chest. Then a small change of the hand configuration has been made to complete the sign accurately where the both hands are located to each other at the same previously defined position.

An open source platform is also available for representing signs using the HamNoSys notation [37]. This platform includes an input palette and five distinct sections, each of which represents different aspects of the sign: handshape, orientation, location, movement, and two-handed notations. The platform provides users with a set of symbols for each of these sections, which enables them to easily create and customize their own signs.

This section provides the symbols to represent handshape.

Figure 4.20: Handshape Section

There are two individual sections to provide Orientation and location symbols.

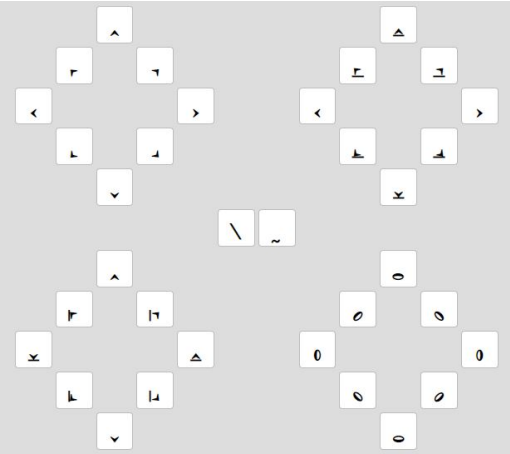

Figure 4.21: Orientation Section

Figure 4.22: Location Section

There are two sections dedicated to movement symbols, with one focused on straight movements and the other for circular movements.

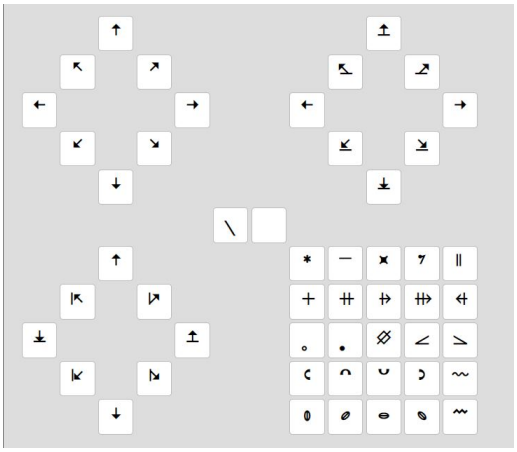

Figure 4.23: Straight Movement Section

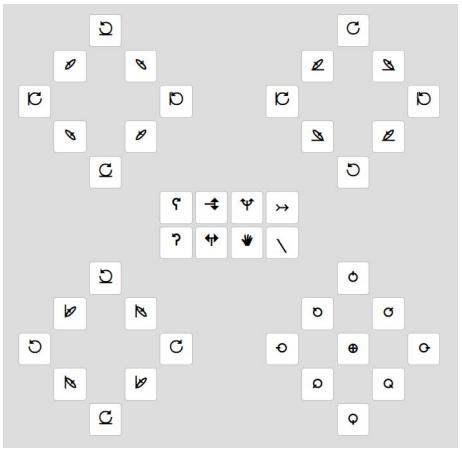

Figure 4.24: Circular Movement Section

The last section offers the symbols necessary to represent two-handed signs.

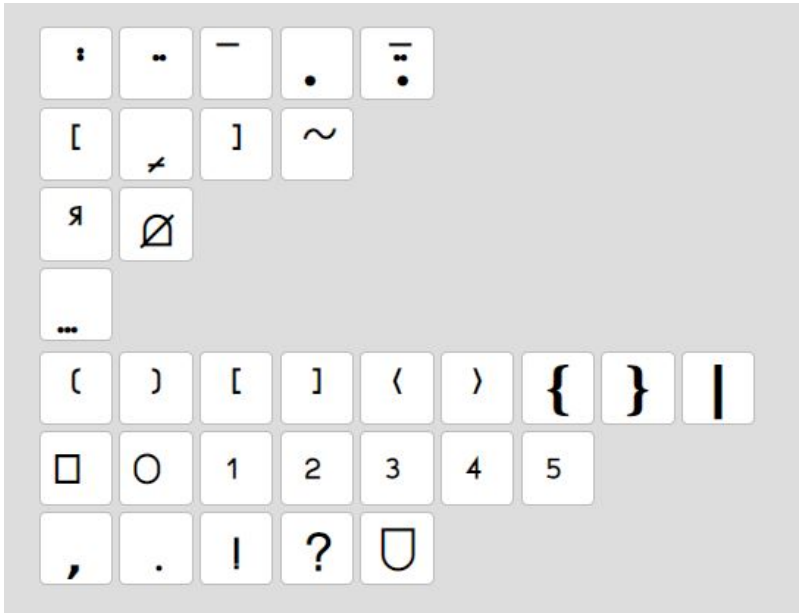

Figure 4.25: Two Handed Section

4.6 HamNoSys to SiGML file generation

SiGML is type of XML file that is used to represent each symbol in HamNoSys for a particular sign. A gloss attribute has a meaning that frequently relates to the word name for which its SiGML is generated. The non-manual features can also be included. The equivalent tags for the non-manual features are placed between the tags *<hamnosys_nonmanual>* and *</hamnosys_nonmanual>* . On the other hand, the tags *<hamnosys_manual>* and *</hamnosys_manual>* are used for manual ones. After generating the appropriate hamnosys notation for a particular word, we need to contruct the SiGML files to animate the sign of the word. Here, we are using "SiS-Builder," a publicly accessible web platform, to create the relevant SiGML files [38]. Each of these SiGML files is quite small (about 2−3 kilobytes in size).

The appropriate SiGML file of the character 'আ' shown in figure 4.26:

```
<sigml>
        <hns_sign gloss="আ">
                <hamnosys_nonmanual/>
                <hamnosys_manual>
                        <hamsymmpar/>
                        <hamparbegin/>
                        <hamfinger2/>
                        <hamthumbacrossmod/>
                        <hamextfingerd/>
                        <hambetween/>
                        <hamextfingerol/>
                        <hampalml/>
                        <hamplus/>
                        <hamfinger2345/>
                        <hamthumboutmod/>
                        <hamextfingero/>
                        <hampalmu/>
                        <hamparend/>
                        <hamparbegin/>
                        <hamfingertip/>
                        <hamthumb/>
                        <hamplus/>
                        <hamfingertip/>
                        <hamindexfinger/>
                        <hamparend/>
                        <hamtouch/>
                        <hamchest/>
                </hamnosys_manual>
        </hns_sign>
</sigml>
```

Figure 4.26: SiGML of আ

The appropriate SiGML file of the digit ০ (শূন্য) shown in figure 4.27

```
<sigml>
        <hns_sign gloss="শূন্য">
                <hamnosys_nonmanual/>
                <hamnosys_manual>
                        <hampinch12/>
                        <hamextfingeru/>
                        <hampalml/>
                        <hamneck/>
                </hamnosys_manual>
        </hns_sign>
</sigml>
```

Figure 4.27: SiGML of the digit ০ (শূন্য)

The appropriate SiGML file of the word ‘বই’ shown in figure 4.28

```
<sigml>
    <hns_sign gloss="বই">
         <hamnosys_nonmanual/>
              <hamnosys_manual>
                      <hamsymmpar/>    <hamflathand/>
                      <hamthumboutmod/>   <hamparbegin/>
                      <hamextfingero/>   <hampalml/>
                      <hamplus/>   <hamextfingero/>
                      <hampalmr/>   <hamparend/>
                      <hamparbegin/>   <hamfingerpad/>
                      <hamindexfinger/>   <hamplus/>
                      <hamfingerpad/>   <hamindexfinger/>
                      <hamparend/>   <hamtouch/>
                      <hamshoulders/>   <hamnomotion/>
                      <hamreplace/>   <hamextfingero/>
                      <hampalmu/>   <hamparbegin/>
                      <hamfingertip/>   <hammiddlefinger/>
                      <hamplus/>    <hamfingertip/>
                      <hammiddlefinger/>    <hamparend/>
                      <hamclose/>     <hamshoulders/>
              </hamnosys_manual>
       </hns_sign>
</sigml>
```

Figure 4.28: SiGML of the word বই

4.7 Animation Creation

Once the SiGML file is received, it is fed into the avatar to create the corresponding animation. To test the SiGML files for a specific word, we utilize a software called SiGML player [39], available for Windows, Mac and Linux operating system. Anna, Marc, Luna etc. are the 3D avatar characters present in this software to show the visual representation of any sign. Moreover, this tool allows us to input the SiGML and view the corresponding animation. In case of incorrect animation, we modify the HamNoSys notation, generate a new SiGML file and feed it back into the SiGML player until the correct avatar animation is obtained. Once the correct animation is achieved, the SiGML file is included in the corpus of our system.

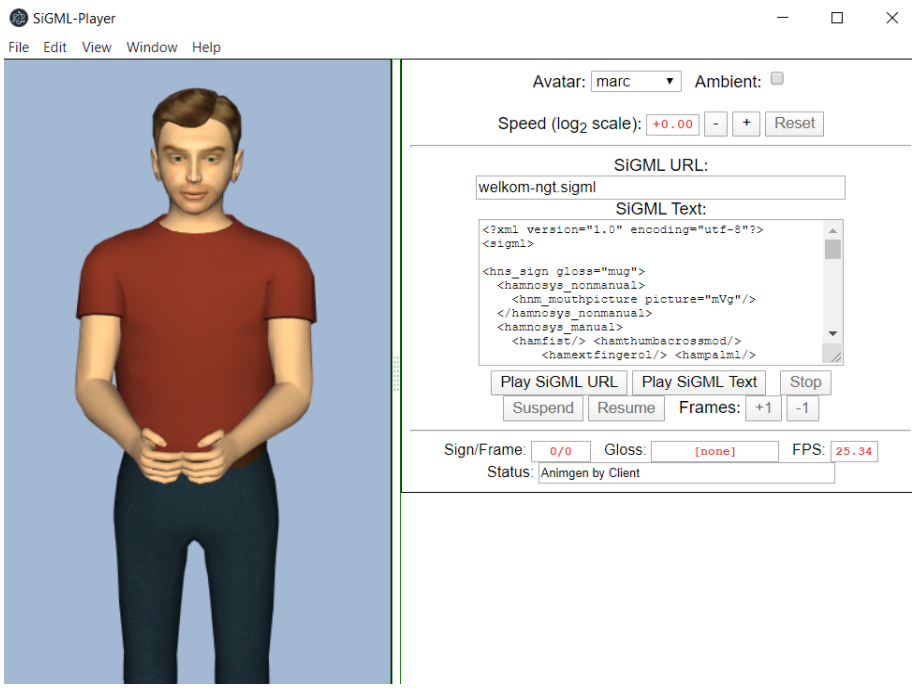

Figure 4.29: SiGML Player

Chapter 5

Animations

For animation we are using JASigning software that provides variety of 3D avatars for generating sign language [40]. The avatar produces the animations frames based on the SiGML file. This way the user can see the sign language of a given word or sentence. As animations are not still pictures, they cannot be added here perfectly. As a result, the final gestures of the corresponding sign have been added here instead.

5.0.1 Digits

The corresponding Avatar-based animated sign of the digit ০ (শূন্য) :

Figure 5.1: Sign of ০ (শূন্য), BDSL sign (*left side*), Avatar animation (*right side*)

The corresponding Avatar-based animated sign of the digit ৭ (সাত):

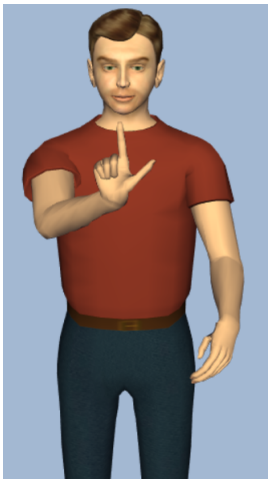

Figure 5.2: Sign of ৭ (সাত), BDSL sign (*left side*), Avatar animation (*right side*)

5.0.2 Characters

The corresponding Avatar-based animated sign of the character 'আ' :

Figure 5.3: Sign of আ, BDSL sign (*left side*), Avatar animation (*right side*)

The corresponding Avatar-based animated sign of the character 'গ':

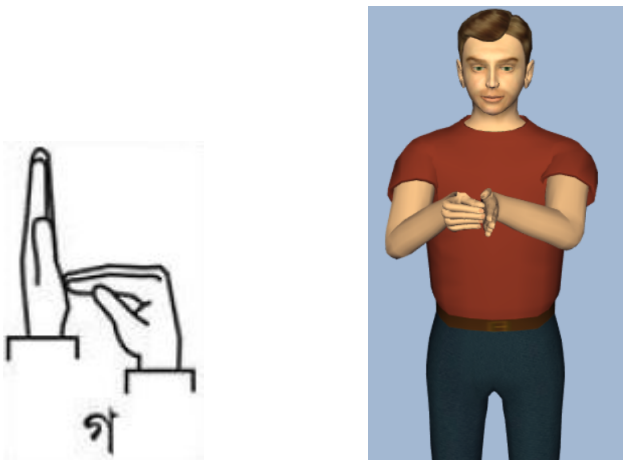

Figure 5.4: Sign of গ, BDSL sign (*left side*), Avatar animation (*right side*)

5.0.3 Words

The corresponding Avatar-based animated sign of the word ‘বই’:

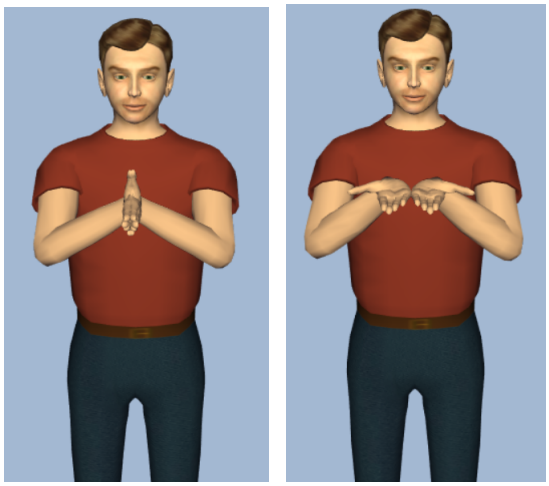

Figure 5.5: Sign of বই, BDSL sign (*Up*), Avatar animation (*Down*)

The corresponding Avatar-based animated sign of the word ‘যাও(অনুরোধ)’:

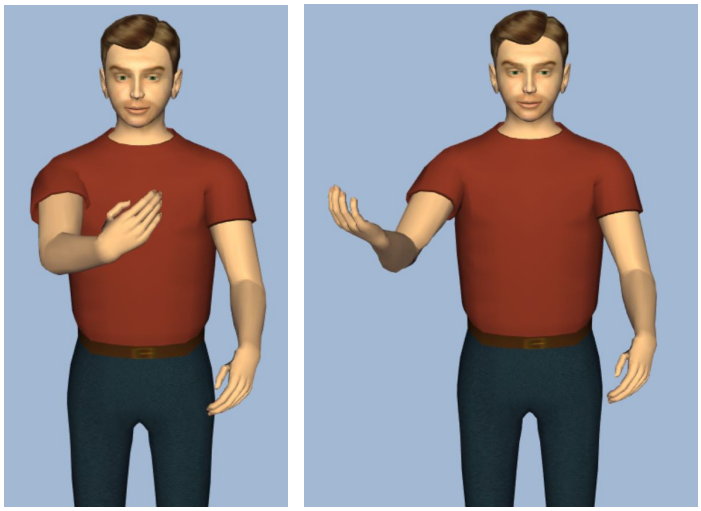

Figure 5.6: Sign of যাও, BDSL sign (*Up*), Avatar animation (*Down*)

The corresponding Avatar-based animated sign of the word ‘কৃত্রিম’:

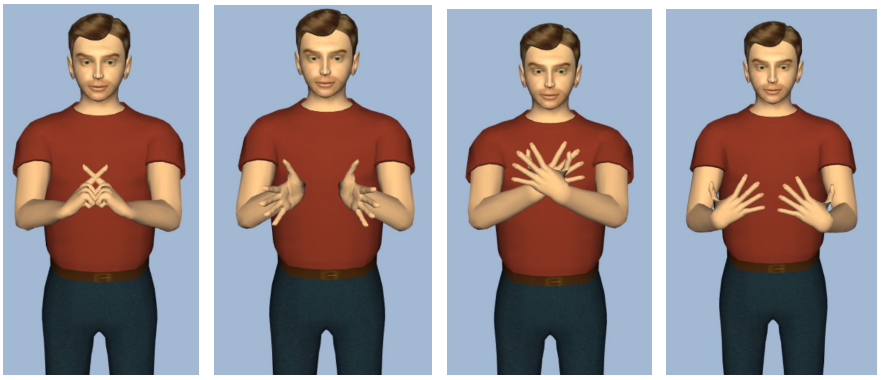

Figure 5.7: Sign of কৃত্রিম, BDSL sign (*Up*), Avatar animation (*Down*)

5.0.4 Sentences

This section is about the demonstration of some Bangla sentences. Since it is not possible to directly show these animations on video, specific and valid movements are shown for each word sequentially. For the first sentence: "আমাদের দেশের নাম বাংলাদেশ", the animated sign of our system and actual sign representation of our BDSL dataset will be:

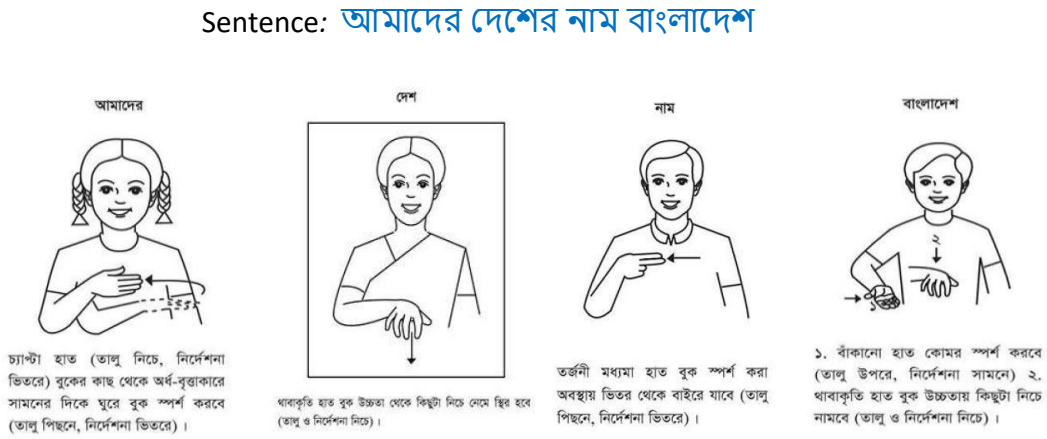

Figure 5.8: Sentence 1 - আমাদের দেশের নাম বাংলাদেশ

Figure 5.9: Animation of Sentence 1 - আমাদের দেশের নাম বাংলাদেশ

This is another animation of example for the sentence, “তার পোষা কুকুরটির রং কালো”, where figure 5.10 is the demonstration of word’s sign that are present in input sentence.

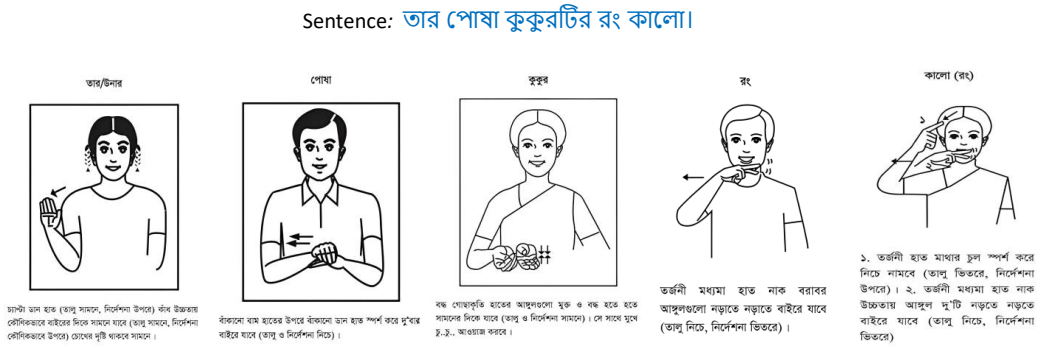

Figure 5.10: Sentence 2 - তার পোষা কুকুরটির রং কালো

The corresponding avatar animation of our system for the sentence:

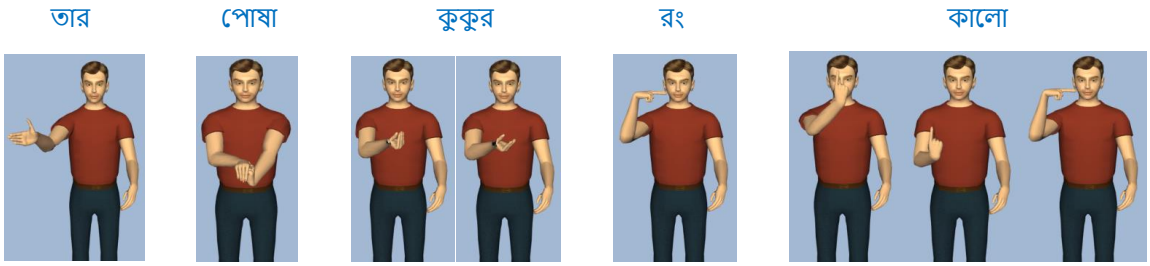

Figure 5.11: Animation of Sentence 2 - তার পোষা কুকুরটির রং কালো

Chapter 6

Interface Design

We have created an interface for displaying the animation of the given input bangla text. The interface contains the avatar along with three sections:

1. **Input section:** For taking bangla text as input from user.

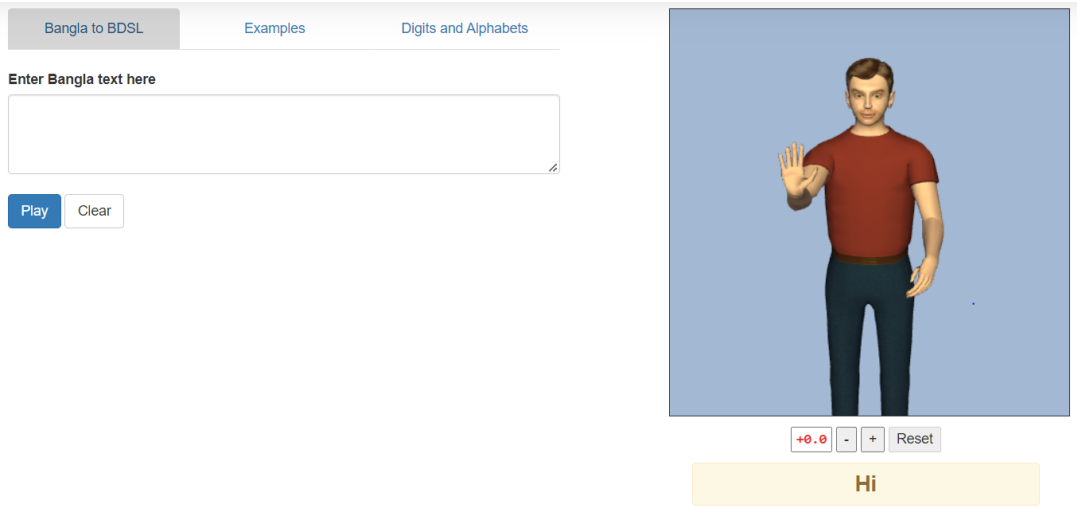

Figure 6.1: Interface - Input section

2. **Example section:** This section contains some predefined commonly used bangla sentences.

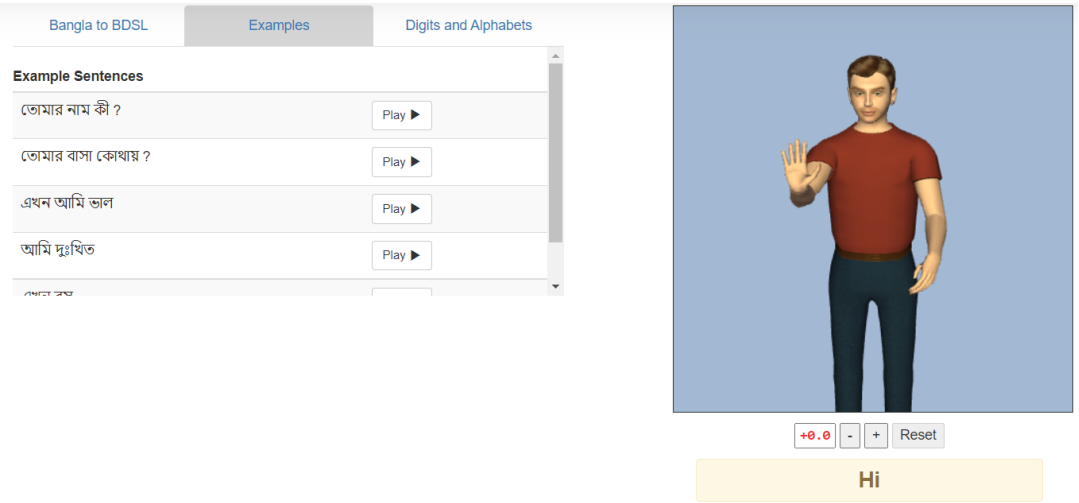

Figure 6.2: Interface - Example section

3. **Digits and alphabets section:** Bangla alphabets and digits are included here.

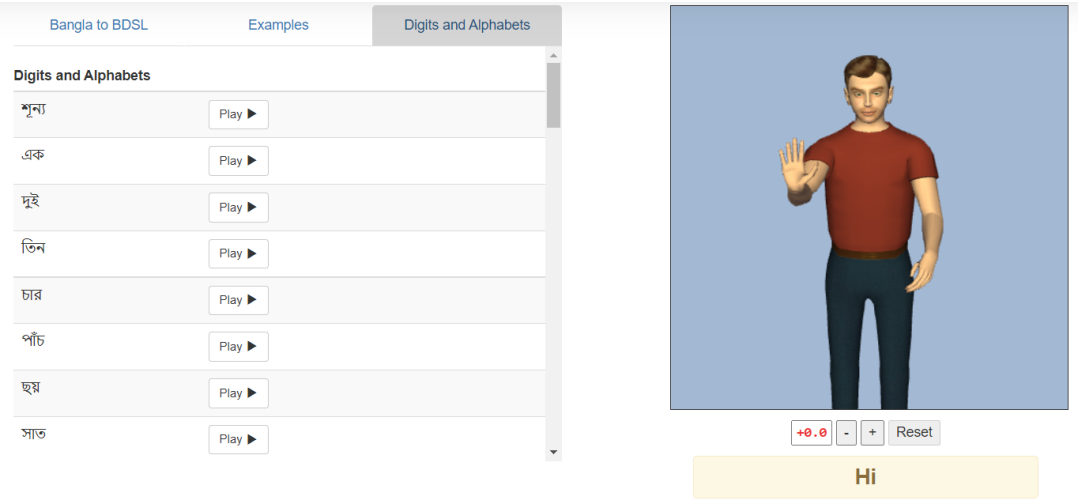

Figure 6.3: Interface - Digits section

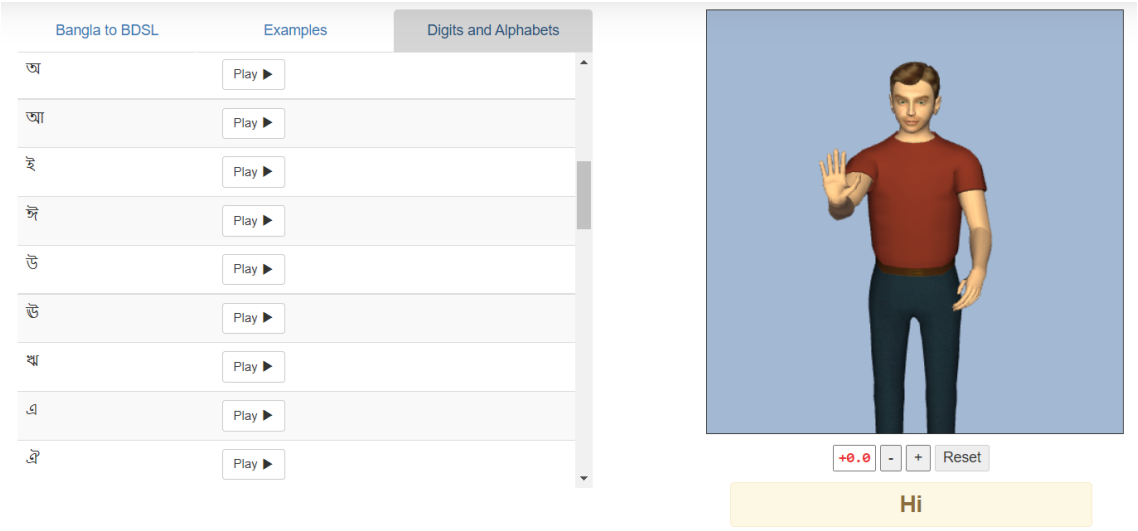

Figure 6.4: Interface - Alphabets section

First the user will enter the text to be animated. After pressing the play button, the avatar will show the desired animation. Moreover, there are two buttons for adjusting the speed of the animation . The avatar animation can also be seen from different angles.

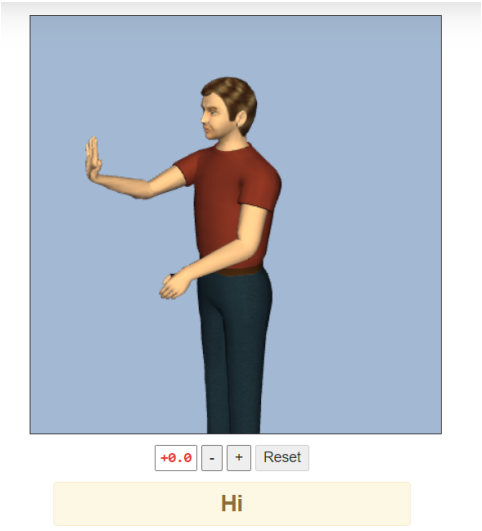

Figure 6.5: Avatar View - Angle 1

Figure 6.6: Avatar View - Angle 2

Chapter 7

HamNoSys-Based BDSL Corpus

The creation of HamNoSys based Bangla Sign Language (BDSL) corpus is one of the fundamental component for our entire system. The corpus has been developed on the basis of BDSL dictionary, provided by the Committee on Standardization of Bangla Language for use in Information Technology (BCC). In addition, this whole dictionary was monitored by he Executive Director of BCC, a statutory body of the ICT Division of the Ministry of Posts, Telecommunications, and IT. The dictionary contains about four thousand sign words that are divided into various categories: পরিবার ও আত্মীয়(Family & Relatives), সম্পর্ক(Relationships), শিক্ষা(Education), কাজ(Job), মানুষের চারিত্রিক বৈশিষ্ট্য(Human Characteristics), প্রকৃতি ও পরিবেশ(Nature & Environment), ভৌগলিক উপাদান(Geographical Elements), রোগ ও চিকিৎসা(Disease & Treatment), খেলাধুলা(Sports), ধর্ম(Religion), প্রাতিষ্ঠানিক শব্দাবলী(Institutional Words), প্রতিবন্ধিতা(Disability), সামাজিক রীতিনীতি ও অনুষ্ঠান(Social Custom & Occasion) etc.

Among four thousand of words, our corpus consists of 2159 sign words. The details of the corpus on the basis of each category is provided in table 7.1.

Index	Category	Corpus
1	অপরাধ ও আইন (Crime and Law)	22
2	অর্থনৈতিক (Economic)	35
3	আন্তর্জাতিক (International)	26
4	উৎসব (Festival)	21
5	ঐতিহাসিক (Historical)	11
6	কর্মী (Worker)	77
7	কাজ (Job)	45
8	খাবার ও পানীয় (Food and Drinks)	88
9	গুরুত্বপূর্ণ স্থান (Important Places)	20
10	গৃহস্থালী সামগ্রী (Household Items)	52
11	দেশ (Country)	32
12	ধর্ম (Religion)	107
13	পরিবহন (Transport)	79
14	পরিবার ও আত্মীয় (Family and Relatives)	40
15	পশুপাখি (Animals and Birds)	73
16	পেশা (Profession)	36
17	পোষাক পরিচ্ছেদ (Clothing)	32
18	প্রকৃতি ও পরিবেশ (Nature and Environment)	101
19	অক্ষর (Letter)	49
20	বাক্য (Sentence)	6
21	ব্যাকরণ (Grammar)	21
22	মানুষের চারিত্রিক বৈশিষ্ট্য (Human Characteristics)	409
23	মানুষের শরীরের অঙ্গ (Human Body Parts)	61
24	রাজনীতি (Politics)	149
25	রোগ ও চিকিৎসা (Disease and Treatment)	35
26	শহর (City)	63
27	শিক্ষা (Education)	52
28	সংখ্যা (Number)	10
29	সামাজিক কাজ (Social Work)	31
30	অন্যান্য (Others)	376
	Total	2159

Table 7.1: Corpus description

Chapter 8

System Evaluation

For system evaluation, we selected text randomly from website [41]. We used 15 short stories as input text, each with very simple sentences. We divided the 15 stories into 5 texts, each containing 3 stories. As there is currently no established evaluation process for sign language generation systems, we assessed the performance of our system by measuring its ability to generate sign language for the five texts provided. In addition, we evaluated how our system handled words that do not have a corresponding sign(অনুসর্গ, অব্যয়). Our system processes texts that contain 30 to 40 sentences and over 300 Bangla words each. To evaluate the system's performance, we measure its ability to generate sign animations based on the lemmas extracted from the texts. The results of our evaluation are presented in the table below, which consists of four columns: Text no, Total Lemmas, Total Unique Lemmas and Total Lemmas Found. The description of each column is given below:

- *Total Lemmas:* The number of root words obtained by the lemmatizer after removing unnecessary words from the text.
- *Total Unique Lemmas:* The number of unique lemmas within the Total Lemmas.
- *Total Lemmas Found:* The number of signs for the unique lemmas that were present in our system.

These findings provide valuable insights into the strengths and limitations of our sign language generation system.

Text No	Total Lemmas	Unique Lemmas	Unique Lemmas Found
1	312	168	97
2	429	202	160
3	347	196	125
4	295	174	109
5	306	182	117

Table 8.1: System Evaluation: Encoder Decoder Based RNN

The two evaluated approaches are encoder decoder based RNN and Word Mapping and Rule based approach which provide outputs based on analysis of five input texts. The Word Mapping and Rule Based approach identified a higher total number of lemmas and unique lemmas compared to the Encoder Decoder based RNN approach. This difference in performance is attributed to the lower accuracy of the Encoder Decoder based RNN in identifying root words or lemmas. Certain words in the input texts fall into an unnecessary category and are not considered as lemmas in the Encoder Decoder based RNN. For instance, the word "তাই/ তার" is lemmatized as "তা" using the Encoder Decoder based RNN, which is not relevant to generating sign animations, and thus, not counted as a lemma. Additionally, the Word Mapping and Rule Based approach performed slightly better in identifying unique lemmas in the input texts than the Encoder Decoder based RNN model.

Text No	Total Lemmas	Unique Lemmas	Unique Lemmas Found
1	328	178	101
2	447	209	172
3	360	197	133
4	310	176	121
5	316	189	119

Table 8.2: System Evaluation: Rule and Mapping Based Approach

Chapter 9

Results and Discussion

As of now, we've completed the animation for 2159 bangla words including all the digits (০-৯) and alphabets (স্বরবর্ণ এবং ব্যঞ্জনবর্ণ). Our system can fully animate a full bangla sentence if all the individual words used in that sentence are present in our system. If a word does not exist in our system, its letters will be animated individually. The created animation is easy to understand and the user has full control of the animation speed if he has any difficulties understanding.

Since animated video cannot be exhibited here properly, the animation of the sentence "তুমি এখানে বস" is shown step-by-step through the equivalent subsequent figures for each of those words.

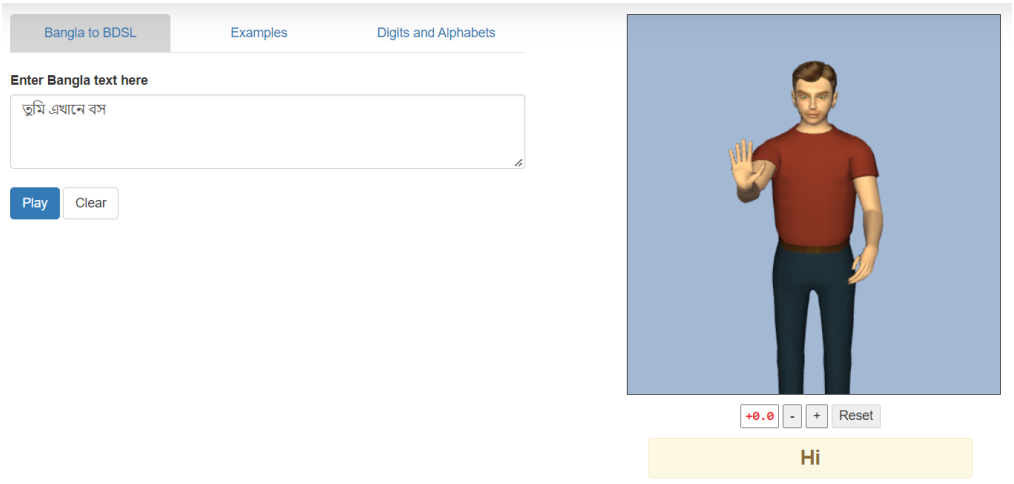

Figure 9.1: Input - তুমি এখানে বস

Figure 9.2: Output animation - তুমি

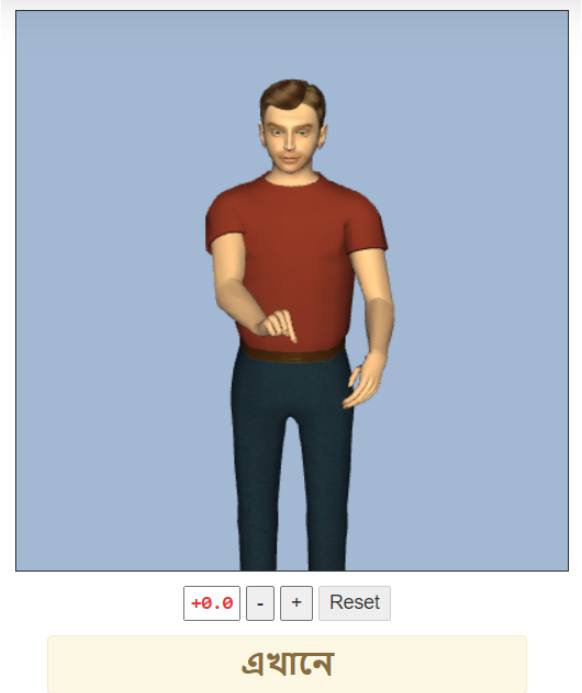

Figure 9.3: Output Animation - এখানে

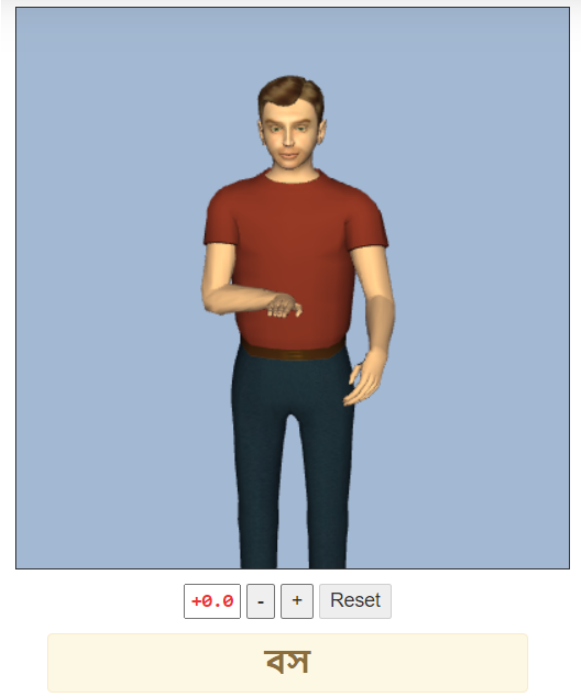

Figure 9.4: Output Animation - বস

Our system has not yet been able to effectively process sentences where some word is too lengthy, as we have used our custom designed lemmatization model to predict all the lemmas. The modification of such smaller units of our system is still undergoing. Despite being a system of automated signs, some of the signs contain various types of body expressions that cannot be represented by a 3D avatar model. However, the overall process of generating Bangla Sign Language from any Bangla text is illustrated in figure 9.5, where the unnecessary word 'নিয়ে' (in perspective of this sentence) has been removed by the parser module while processing the input "আমরা ইশারা ভাষা নিয়ে কাজ করছি" (We are working on sign language).

Sentence: আমরা ইশারা ভাষা নিয়ে কাজ করছি।

Parser (Predicted Lemmas): আমরা ইশারা ভাষা কাজ করা

Sigml Files:

```
<hns_sign gloss="আমরা">
<hamnosys_nonmanual/>
<hamnosys_manual>
<hamfinger2/>
<hamthumbacrossmod/>
<hamextfingerur/>
<hambetween/>
<hamextfingero/>
<hampalml/>
<hamshoulders/>
<hamparbegin/>
<hamcircleu/>
<hamlargemod/>
<hamreplace/>
<hamextfingeri/>
<hampalmd/>
<hamshoulders/>
<hamtouch/>
<hamparend/>
</hamnosys_manual>
</hns_sign>
```

```
<hns_sign gloss="ইশারা">
<hamnosys_nonmanual/>
<hamnosys_manual>
<hamsymmpar/>
<hamflathand/>
<hamthumboutmod/>
<hamextfingero/>
<hampalml/>
<hampalm/>
<hamclose/>
<hamparbegin/>
<hamseqbegin/>
<hamcirclel/>
<hamclocku/>
<hamseqend/>
<hamplus/>
<hamseqbegin/>
<hamcirclel/>
<hamclockd/>
<hamseqend/>
<hamparend/>
<hamrepeatfromstart/>
</hamnosys_manual>
</hns_sign>
```

```
<hns_sign gloss="ভাষা">
<hamnosys_nonmanual/>
<hamnosys_manual>
<hamceeall/>
<hamextfingero/>
<hambetween/>
<hamextfingerr/>
<hampalml/>
<hamshouldertop/>
<hamlrat/>
<hamclose/>
<hammoved/>
</hamnosys_manual>
</hns_sign>
```

```
<hns_sign gloss="কাজ">
<hamnosys_nonmanual/>
<hamnosys_manual>
<hamsymmpar/> <hamfist/>
<hamparbegin/> <hamextfingerol/>
<hambetween/> <hamextfingerl/>
<hampalml/> <hamplus/>
<hamextfingeror/> <hampalmr/>
<hamparend/> <hamparbegin/>
<hamshouldertop/> <hambetween/>
<hamshoulders/> <hamlrat/>
<hamplus/> <hamchest/>
<hamparend/> <hamparbegin/>
<hamparbegin/> <hammovedl/>
<hamsmallmod/> <hammovedl/>
<hamsmallmod/> <hamparend/>
<hamplus/> <hamnomotion/>
<hamparend/> <hamrepeatfromstart/>
</hamnosys_manual>
</hns_sign>
```

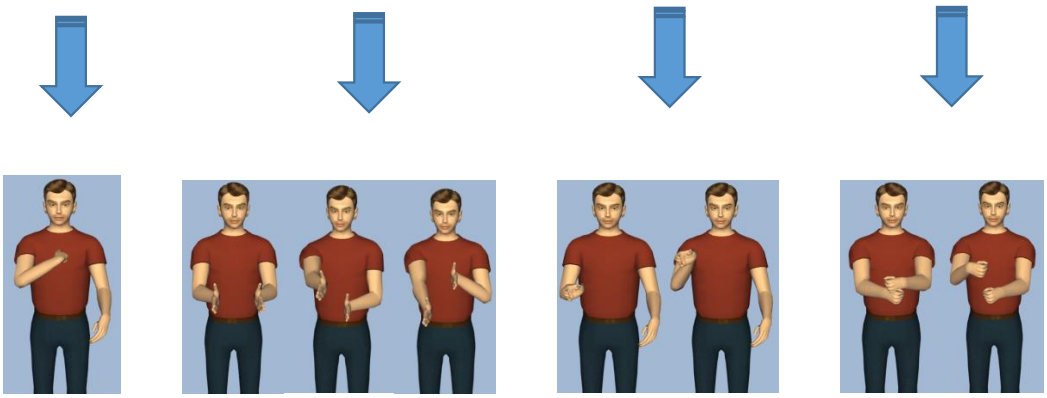

Figure 9.5: Full Overview of Generating Sign Language Animation.

Chapter 10

Error Analysis

It has been observed that the proposed system faces certain limitations in depicting complex signs in BDSL, which require extensive animations. This is due to the limited notations available in HamNoSys. Moreover, The proposed system experiences delays in producing animation for complex words, resulting in slower output. It is also noteworthy that the system's parser is dependent on morphological information, which may sometimes result in incorrect mappings and finding wrong lemmas, leading to sentences that do not have any meaning. Additionally, the system currently lacks the ability to handle complex sentences.

Chapter 11

Limitations

Resource development and benchmarking have been important issues for a low-resource and morphologically complex language like Bangla. Moreover, the domain of Bangla sign language is also very complex and diverse. Despite of having these realities, our system has certain limitations:

- **Corpus:** Despite having a dataset of approximately 4000 sign words, our system has the coverage only 2200 of them (approximately). Hence, the vocabulary size of our system is not particularly large. As a result, expanding the size of our hamnosys-based corpus is currently ongoing.

- **Lemmatizatizer:** To obtain the underlying lemmas in each Bangla text and create the appropriate sign, we used our custom built lemmatization model. Though our model is approximately 80% accurate in this regard, it is unable to correctly predict the lemmas of any lengthy word (approximately more than 12 characters). In fact, a well-known lemmatizer (e.g., the BaNeL lemmatizer) provided 95.75% accuracy, we intend to use it in our system to identify lemmas more effectively and correctly.

- **Word Sense Disambiguation:** While parsing any Bangla text, identifying the right meaning of a word in the given context (known as Word Sense Disambiguation, WSD) is also one of the major tasks of the parser so that the system can generate the actual sign language for the given input text. This is a significant issue in natural language processing since many words in a language might have different meanings depending on the context. Consider the following two sentences: 'উপর থেকে পানি পড়ছে'(Water is falling from the top), 'আমি বই পড়ছি'(I am reading the book). Here, the lemma for the word 'পড়ছে' and 'পড়ছি' is: 'পড়া' in both cases. But the actual meaning of these

words are completely different. We currently have not yet incorporated a solution for this type of problem into our system.

- **Unnecessary words selection:** In the context of the Bangla language, not all words in a sentence have individual meanings. Therefore, for certain words such as (অনুসর্গ, অব্যয়বাচক শব্দ), there are no actual meaningful signs in Bangla sign language. Our system currently identifies only a limited number of such insignificant words.

- **Synonym words:** Replacing words with the related ones might be an effective technique for enhancing the performance of our system. Consider the following two sentences: 'আমার বাসা সিলেটে'(My home is in Sylhet) & 'আমার বাড়ি সিলেটে'(My home is in Sylhet). Here the meaning of both of the words: 'বাসা' and 'বাড়ি' is exactly same, as each of them is synonym to other. We have sign available for the word 'বাসা' in our corpus. On the other hand, there is no sign available for the word 'বাড়ি'. To overcome this issue, we need a large dataset of all possible synonym words of all sign words that are present in our corpus. Hence, utilizing synonyms or similar words can enrich our system as well as making it more stable for generating Bangla Sign language.

- **Sign Difficulty:** While creating our HamNoSys-based corpus, we encountered difficulty in representing certain directional signs for some specific words. For instance, when indicating the sign for the word 'প্যান্ট', it requires raising both hands up to the waist while touching the thighs, which can be quite challenging due to the need to touch the lower parts of the body, and as well as touching the backside of any body parts.

Chapter 12

Future Work

In our future work, we will address the limitations of our current system. Firstly, we will include all 4000 available signs in our dataset to ensure maximum coverage. Secondly, we will work on improving the accuracy of our lemmatizer by training it with more data and utilizing an attention-based encoder-decoder RNN. Thirdly, we will conduct further research to identify unnecessary words that have no available signs.

To handle complex signs and tenses, we plan to incorporate additional rules into our system. Additionally, we will work on word sense disambiguation to ensure accurate interpretation of sign language. For proper evaluation of our system, we will engage professional interpreters to provide feedback on its overall performance. Moreover, we will implement an avatar wearing our cultural outfit for sign animations. These efforts will allow us to improve the efficiency and acceptability of our system.

Chapter 13

Conclusion

Our system enables the translation of Bangla text into sign language. Previous works on generating Bangla sign language were limited to numbers, alphabets and few words. These earlier works used pre-recorded videos for sign visualization and did not cover full Bangla sentence animation. To our knowledge, there is currently no fully built-in system available that can generate Bangla sign language like ours. Our approach is highly memory-efficient, as it does not require pre-recorded videos for animation. Furthermore, our 3D avatar animation is generated in real-time, allowing for clear visualization of the sign created by the avatar. Our system is expected to greatly assist people in learning and understanding Bangla Sign Language. If we include all available sign words and complete the text processing part, we hope that our proposed system will be an effective means of communication with hearing-impaired individuals.

References

[1] BSTI, "Specification of Bangla Sign Language (BDSL) (First Version)," July 2019.

[2] R. Smith, "Hamnosys," url : https://robertsmithresearch.files.wordpress.com/2012/10/hamnosys-user-guide-rs-draft-v3-0.pdf , Available [Online].

[3] T. Hanke, "Hamnosys-representing sign language data in language resources and language processing contexts," in *LREC*, vol. 4, 2004, pp. 1–6.

[4] W. H. Organization. Deafness and hearing loss. [Online]. Available: https://www.who.int/news-room/fact-sheets/detail/deafness-and-hearing-loss

[5] R. Campbell, M. MacSweeney, and D. Waters, "Sign language and the brain: a review," *The Journal of Deaf Studies and Deaf Education*, vol. 13, no. 1, pp. 3–20, 2008.

[6] P. Kumar and S. Kaur, "Sign language generation system based on indian sign language grammar," *ACM Transactions on Asian and Low-Resource Language Information Processing (TALLIP)*, vol. 19, no. 4, pp. 1–26, 2020.

[7] M. Shamsi, M. Divani, and A. Rasouli Kenari, "Designing an avatar-based translator system from persian into persian sign language (psl)," *Technology of Education Journal (TEJ)*, vol. 15, no. 2, pp. 277–290, 2021.

[8] T. Oliveira, P. Escudeiro, N. Escudeiro, E. Rocha, and F. M. Barbosa, "Automatic sign language translation to improve communication," in *2019 IEEE Global Engineering Education Conference (EDUCON)*. IEEE, 2019, pp. 937–942.

[9] M. T. Hoque, M. Rifat-Ut-Tauwab, M. F. Kabir, F. Sarker, M. N. Huda, and K. Abdullah-Al-Mamun, "Automated bangla sign language translation system: Prospects, limitations and

applications," in *2016 5th International Conference on Informatics, Electronics and Vision (ICIEV)*. IEEE, 2016, pp. 856–862.

[10] G. F. S. Eberhard, David M. and C. D. Fennig, "Ethnologue: Languages of the world." twenty-sixth edition. Dallas, Texas: SIL International. [Online]. Available: https://www.ethnologue.com/language/ben

[11] A. Chakrabarty and U. Garain, "Benlem (a bengali lemmatizer) and its role in wsd," *ACM Transactions on Asian and Low-Resource Language Information Processing (TALLIP)*, vol. 15, no. 3, pp. 1–18, 2016.

[12] T. Ozawa, "An integral image and text processing system for automatic generation of 3d sign-language animations," in *ISCAS 2001. The 2001 IEEE International Symposium on Circuits and Systems (Cat. No. 01CH37196)*, vol. 2. IEEE, 2001, pp. 313–316.

[13] M. Papadogiorgaki, N. Grammalidis, D. Tzovaras, and M. G. Strintzis, "Text-to-sign language synthesis tool," in *2005 13th European Signal Processing Conference*. IEEE, 2005, pp. 1–4.

[14] B. Sarkar, K. Datta, C. Datta, D. Sarkar, S. J. Dutta, I. D. Roy, A. Paul, J. U. Molla, and A. Paul, "A translator for bangla text to sign language," in *2009 Annual IEEE India Conference*. IEEE, 2009, pp. 1–4.

[15] J. Porta, F. López-Colino, J. Tejedor, and J. Colás, "A rule-based translation from written spanish to spanish sign language glosses," *Computer Speech & Language*, vol. 28, no. 3, pp. 788–811, 2014.

[16] M. Varghese and S. K. Nambiar, "English to sigml conversion for sign language generation," in *2018 International Conference on Circuits and Systems in Digital Enterprise Technology (ICCSDET)*. IEEE, 2018, pp. 1–6.

[17] S. Stoll, N. C. Camgoz, S. Hadfield, and R. Bowden, "Text2sign: towards sign language production using neural machine translation and generative adversarial networks," *International Journal of Computer Vision*, vol. 128, no. 4, pp. 891–908, 2020.

[18] M. Sanaullah, B. Ahmad, M. Kashif, T. Safdar, M. Hassan, M. H. Hasan, and N. Aziz, "A real-time automatic translation of text to sign language," *Computers, Materials and Continua*, vol. 70, no. 2, pp. 2471–2488, 2022.

[19] R. RKDMP, W. Wijekoon, K. Rajapakse, K. Rasanjalee, and D. De Silva, "Real-time sign language translator," *International Journal of Engineering and Management Research*, vol. 12, no. 6, pp. 117–124, 2022.

[20] J. Joy, K. Balakrishnan *et al.*, "A prototype malayalam to indian sign language (isl) automatic translator." *International Journal of Next-Generation Computing*, vol. 13, no. 2, 2022.

[21] S. Paul, N. Joshi, and I. Mathur, "Development of a hindi lemmatizer," *arXiv preprint arXiv:1305.6211*, 2013.

[22] D. Zeman, J. Hajic, M. Popel, M. Potthast, M. Straka, F. Ginter, J. Nivre, and S. Petrov, "Conll 2018 shared task: Multilingual parsing from raw text to universal dependencies," in *Proceedings of the CoNLL 2018 Shared Task: Multilingual parsing from raw text to universal dependencies*, 2018, pp. 1–21.

[23] S. L. Ingólfsdóttir, H. Loftsson, J. F. Daðason, and K. Bjarnadóttir, "Nefnir: A high accuracy lemmatizer for icelandic," *arXiv preprint arXiv:1907.11907*, 2019.

[24] B. Jongejan and D. Haltrup, "the cst lemmatiser," *Center for Sprogteknologi, University of Copenhagen version*, vol. 2, 2005.

[25] A. K. Ingason, S. Helgadóttir, H. Loftsson, and E. Rögnvaldsson, "A mixed method lemmatization algorithm using a hierarchy of linguistic identities (holi)," in *Advances in Natural Language Processing: 6th International Conference, GoTAL 2008 Gothenburg, Sweden, August 25-27, 2008 Proceedings*. Springer, 2008, pp. 205–216.

[26] I. Akhmetov, A. Pak, I. Ualiyeva, and A. Gelbukh, "Highly language-independent word lemmatization using a machine-learning classifier," *Computación y Sistemas*, vol. 24, no. 3, pp. 1353–1364, 2020.

[27] I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to sequence learning with neural networks," *Advances in neural information processing systems*, vol. 27, 2014.

[28] P. Qi, Y. Zhang, Y. Zhang, J. Bolton, and C. D. Manning, "Stanza: A python natural language processing toolkit for many human languages," *arXiv preprint arXiv:2003.07082*, 2020.

[29] K. Milintsevich and K. Sirts, "Enhancing sequence-to-sequence neural lemmatization with external resources," *arXiv preprint arXiv:2101.12056*, 2021.

[30] A. Z. M. Faridee and F. Tyers, "Development of a morphological analyser for bengali," in *Proceedings of the First International Workshop on Free/Open-Source Rule-Based Machine Translation*, 2009.

[31] A. R. Pal, N. S. Dash, and D. Saha, "An innovative lemmatization technique for bangla nouns by using longest suffix stripping methodology in decreasing order," in *2015 International Conference on Computing and Network Communications (CoCoNet)*. IEEE, 2015, pp. 675–678.

[32] A. Chakrabarty, A. Chaturvedi, and U. Garain, "A neural lemmatizer for bengali," in *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, 2016, pp. 2558–2561.

[33] M. Kowsher, I. Hossen, and S. Ahmed, "Bengali information retrieval system (birs)," *International Journal on Natural Language Computing (IJNLC)*, vol. 8, no. 5, 2019.

[34] A. R. Pal, D. Saha, S. K. Naskar, and N. S. Dash, "In search of a suitable method for disambiguation of word senses in bengali," *International Journal of Speech Technology*, vol. 24, pp. 439–454, 2021.

[35] M. A. Islam, M. Towhiduzzaman, M. T. I. Bhuiyan, A. A. Maruf, and J. A. Ovi, "Banel: an encoder-decoder based bangla neural lemmatizer," *SN Applied Sciences*, vol. 4, no. 5, p. 138, 2022.

[36] M. Schulder and T. Hanke, "HamNoSys in TeX," DGS-Korpus project, IDGS, Universität Hamburg, Hamburg, Germany, Project Note AP04-2021-02, 2021. [Online]. Available: https://www.sign-lang.uni-hamburg.de/dgs-korpus/arbeitspapiere/DGS-Korpus_AP04-2021-02v01_en.pdf

[37] "HamNoSys Input — sign-lang.uni-hamburg.de," https://www.sign-lang.uni-hamburg.de/hamnosys/input/, [Accessed 02-Mar-2023].

[38] T. Goulas, S.-E. Fotinea, E. Efthimiou, and M. Pissaris, "Sis-builder: a sign synthesis support tool," in *sign-lang@ LREC 2010*. European Language Resources Association (ELRA), 2010, pp. 102–105.

[39] "CWASA vhg2023 Open — vhg.cmp.uea.ac.uk," https://vhg.cmp.uea.ac.uk/tech/jas/vhg2023/index.html, [Accessed 02-Mar-2023].

[40] R. Elliott, J. Bueno, R. Kennaway, and J. Glauert, "Towards the integration of synthetic sl animation with avatars into corpus annotation tools," in *sign-lang@ LREC 2010*. European Language Resources Association (ELRA), 2010, pp. 84–87.

[41] E. Desk, "ছোটদের ১৫ টি নীতিমূলক গল্প – Top 15 Moral Stories for Kids in Bengali | BongQuotes — bongquotes.com," https://bongquotes.com/bengali-moral-stories-for-kids/, [Accessed 02-Mar-2023].