INDIAN JOURNAL OF SCIENCE AND TECHNOLOGY

RESEARCH ARTICLE

AI Enabled Text-to-Speech Synthesis for Unicode Language

Chandamita Nath[1]*, Bhairab Sarma[2]

1 Research Scholar, Department of Computer Science, University of Science & Technology, Meghalaya, India
2 Associate Professor, Department of Computer Science, University of Science & Technology, Meghalaya, India

OPEN ACCESS

Received: 14-08-2024

Accepted: 05-11-2024

Published: 25-11-2024

Citation: Nath C, Sarma B (2024) AI Enabled Text-to-Speech Synthesis for Unicode Language. Indian Journal of Science and Technology 17(42): 4454-4461. https://doi.org/10.17485/IJST/v17i42.2645

* **Corresponding author**.

dipudoili99@gmail.com

Funding: None

Competing Interests: None

Copyright: © 2024 Nath & Sarma. This is an open access article distributed under the terms of the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original author and source are credited.

Published By Indian Society for Education and Environment (iSee)

ISSN
Print: 0974-6846
Electronic: 0974-5645

Abstract

Objectives: To explore the advancements and challenges in the implementation of an AI-enabled Text-to-Speech (TTS) for the Assamese language (an Indo-Aryan language) by discussing technological approaches, linguistic considerations, and its potential applications. **Methods:** The developed system has been experimented with 40K (approx) collected words of five different categories and tested with different datasets for each model. Four prominent methods (Dictionary, HMM, CNN & G2P) are commonly found used in TTS. With the Grapheme-to- Phoneme (G2P) conversion technique, all phonemes are ordered sequentially to generate a continuous voice by using a good synthesizer to translate into words, letters, ligatures, and symbols of the language into sound. **Findings:** Compared to the other three approaches, the G2P approach shows higher performance (89%) in terms of accuracy. We experimented with our system using different sample sizes from three Indian languages (Hindi, Assamese, and Bengali) across four approaches. With a dataset of 1,000 words, the Dictionary-based approach resulted in 60% accuracy, HMM in 70%, CNN in 85%, and remarkably, the G2P approach reported 89% accuracy. Upon uploading the input file into the system and listening to the system read words, we manually calculated the number of words spelled correctly, incorrectly, or not producing any sound. Precision, F1-score, and recall were analyzed and found to achieve a 90% result, which may be considered a satisfactory conversion. Furthermore, by giving additional thought to pronunciation generation, the accuracy level might be improved. For experimental purposes, only female voices were used in this study. The smaller database size and minimal word decomposition algorithms utilized in this study account for the improved accuracy. **Novelty:** This effort offers a unique solution for the Assamese language. It has been designed with a pleasant voice and is in interactive mode which wais not found earlier.

Keywords: AI; Assamese Language; NLP; G2P; Text-To-Speech

1 Introduction

The goal of a text to speech synthesis (TTS) system aims to generate natural and comprehensible speech from the provided text. Under NLP, numerous works in both Indian and non-Indian languages have been developed. However, very less amount of work has been developed in regional languages, especially in the Assamese Language due to a lack of awareness. The highly inflectional structure and intricate grammatical rules of Indian languages make feature selection and extraction incredibly challenging. However, sentence construction conventions also differ among languages, geographical areas, etc. Different dialects of different languages have different tunes. Additionally, morphology and syntactical category also extend their bounds. In certain situations, a subject, verb, and object combination may take on distinct forms. The most intriguing thing is that many structures can be used to preserve contextual information. Numerous such crucial elements that must be taken into account for feature extraction are covered in this study.

AI-based TTS is more similar to human-based TTS, except for the acoustic generation module. All preprocessing tasks must be compiled with a manual system where a machine learning approach (deep learning) is applied to develop the system to be more realistic. The overview of the proposed system is given in Figure 1.

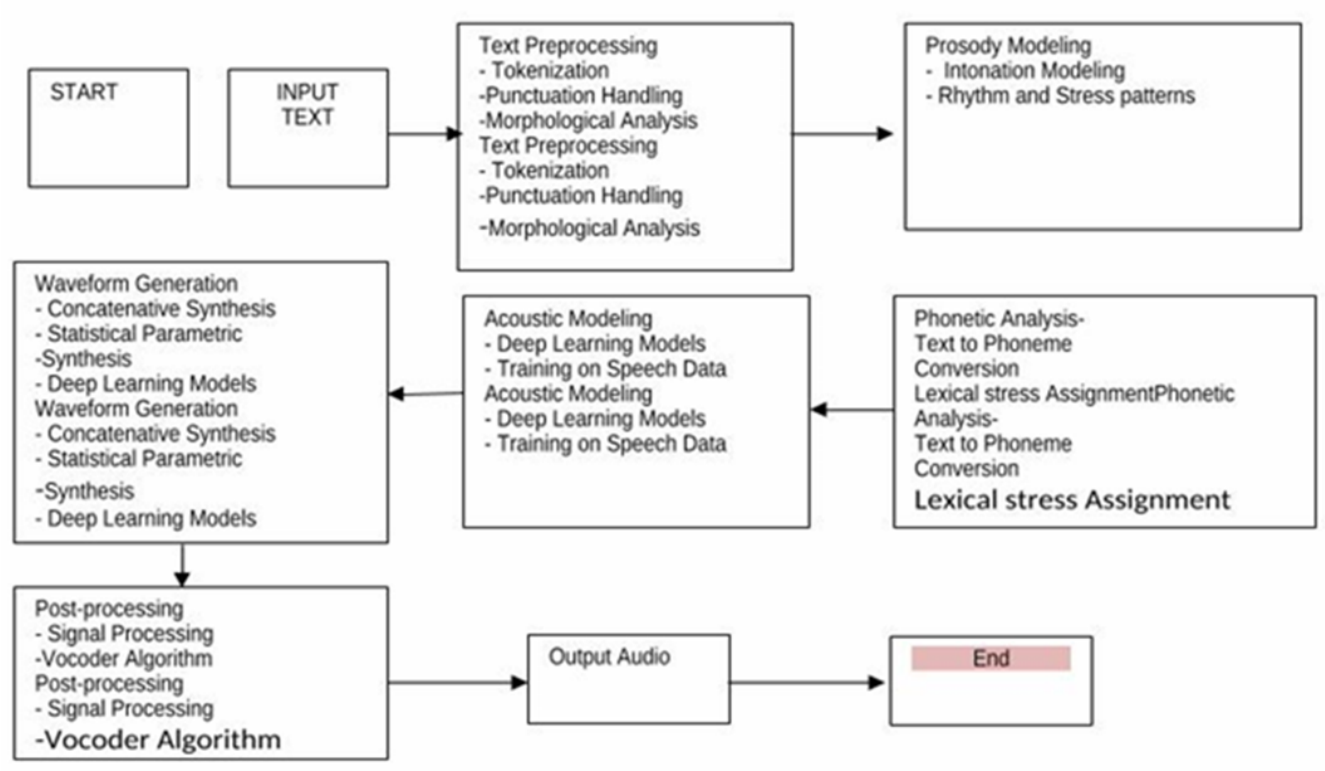

Fig 1. System Overview of the proposed research

Before rolling ahead to TTS, input text must be formatted with a text processing system that includes tokenization and pruning for the removal of spurious attributes and unnecessary features. All special characters, symbols, punctuation markers, and other rich text with smaller pictorial representation will be extracted in the text processing phase. It has been reviewed that many authors experimented with their research with different approaches. It has been observed that TTS is done by using four approaches and they are- Dictionary based approach [(1)]. Hidden Markov model (HMM) [(2)], Convolutional Neural Network (CNN) [(3)] and Grapheme to Phoneme (G2P) [(4)] conversion approach.

Word-by-word conversion methods, often known as dictionary-based approaches, are commonly employed in cases where very short messages need to be converted into speech, such as OTP pronunciation or voice message reading. This method suffers from limited database size and domain dependency. A novel method has been implemented to improve speed and make the TTS system domain-independent, i.e., using graphemes, the smallest units of written text, and phonemes, the smallest units of sound. The fixed size of the database and the formation of words from graphemes to further spell them with a pleasant voice

is the primary goal of this research. There are many challenges in TTS conversion in regional languages. This research is an attempt to expedite the work on regional languages, which are coded in a 16-bit format.

2 Methodology

As discussed, with three prominent approaches, pros and cons have been discovered with respect to the dataset adhered. Out of these, G2P (Grapheme to Phoneme) conversion demanded more convenience and higher accuracy. It is a process that looks, memorizes and recalls a part of written text and converts it into sound [5]. The following section has discussed G2P approaches and their practical implications in traditional system.

2.1 Grapheme Extraction in Unicode

The smallest unit of a printed text is called a grapheme. Each letter in the English language is referred to as a grapheme, and each phoneme is pronounced differently. When a word in Indian language is broken down into its phonemes, several graphemes may emerge. This is because the language's grammatical structure prevents the word from being spoken correctly. In this work, phonemes are denoted as /a/, /b/, etc., and the phonetic transcriptions will be denoted as [a], [b]. Symbolically, graphemes are frequently notated as <a, <b> [6] etc. With this syntax, Grapheme analysis of an input Assamese sentence is explained in Table 1.

Table 1. Grapheme analysis of Assamese text Input sentence

Input sentence	ব্রহ্মপুত্রঅসমৰএখনবৃহৎনদী
Grapheme	<ব><ৰ><হ><ম><প><উ><ত><ৰ><এ><খ><ন><অ><স><ম><ৰ><ৰ><ম্ব><হ><ৎ><ন><দ><ঈ>
Phoneme:	/ba/ /ra/ /ha/ /ma/ /pa/ /u/ /*t/ /ra/I—I /a/ /xa/ /ma/ /r/ I—I /ae/ /xn/ /brue/ /ha/ /~t/ /na/ /dE/
Phonemes acoustic	<ব><ৰ><হ><ম><প><উ><ত><ৰ><এ><খ><ন><অ><স><ম><ৰ><ৰ><ম্ব><হ><ৎ><ন><দ><ঈ>
Assamese sound mapping	/ব্রহ / /ম/ /পুত/ /ৰ/ /-/ /অ/ /স/ /ম/ /~ৰ//-/ / এ/ /খ/ /~ন//-/ /ব্রী// /হ/ /~ত/ /-//ন//অ//দী/

The gap (silent) created by whitespace in input sentences is represented by the phoneme /-/. As given in the table, the word 'bhahmaputra', can be heard as /barahama-p-u-ta-ra/ or /brah-ma-put-ra/ when converted to an acoustic sequence, with an impression following /brah/. This impression was created during the breakdown of graphemes. One encoder-decoder architecture has been used with a supervised approach. In this structure, each input sequence is represented by the encoder as a text (Grapheme), and the decoder uses the learnt representation to produce an output sequence in the.wav format (Phoneme). The block diagram of this process model is as given in Figure 2.

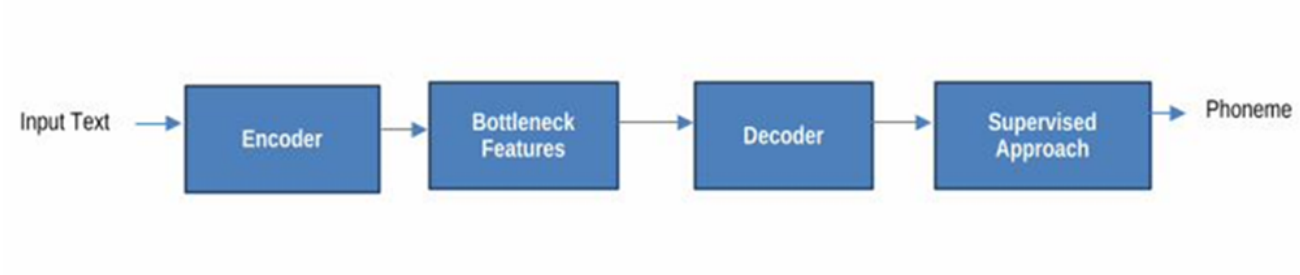

Fig 2. Block diagram of G2P

2.2 Encoder- Decoder

The primary concept of the encoder-decoder techniques consists of two stages: first, the input sequence is mapped to a vector, and then, using the learnt vector representation, the output sequence is generated. After the encoder processes the entire input sequence, encoder-decoder models produce an output that allows the decoder to learn from any portion of the input without being constrained by fixed context windows [7]. Each phoneme is entered into an encoder once the decoded grapheme from the input text has been input into the phoneme creation phase. Based on the learnt representation, the encoder used bottleneck features to encode the output phonemes, which are in the.wav format (Phoneme), to produce continuous audio. Morphological dissemination of an Assamese word **[kitaap]**"কিতাপ" is explained in Figure 3. From the given word, how a phonetic conversion has accomplished is explained below.

In the initial phase of our system, graphemes are decomposed with a small program coded with Python, and phonetic syntheses were composed with the help of audio mixing software (**'FL Studio 20'**used in this experiment). In level 1, each character of the word decomposed with an iteration. Here, the output is termed as grapheme for each character that disseminating with punctuation markers.

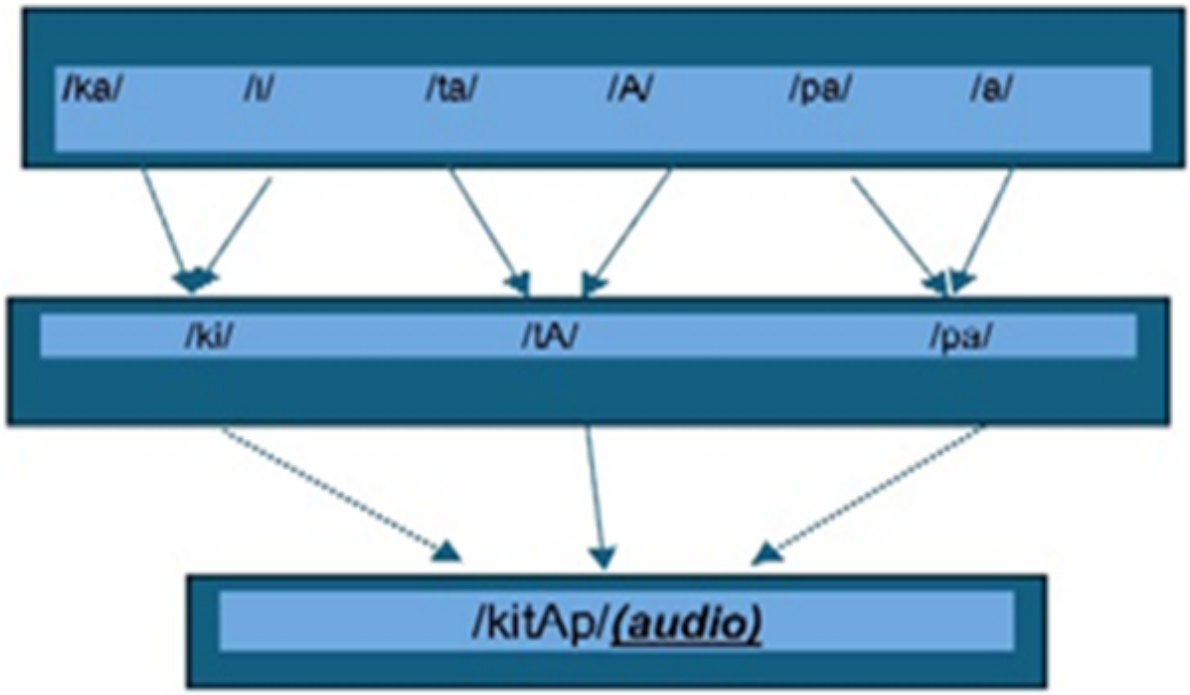

Fig 3. Morphological dissemination of Assamese words

2.3 Grapheme extraction difficulties

Tokenization is used to extract individual words from a phrase. Following tokenization, a loop structure breaks down each distinct word to extract the character before recovering the graphemes. It is possible to extract a single character in a single iteration; however, complicated characters could need more than one iteration[8]. To determine the real graphemes, composition-decomposition techniques are utilized. These issues emerge because they are inflectional. Table 2 summarizes a few observations.

Table 2. Complexity of Grapheme formation due to inflection of secondary form of vowels

Inflections	Composition	Decomposition
Consonant with Vowel	K+ A->kA (ক + আ→কা) K+I->kI(ক + ঈ→কী)	kA->k+.... (not a valid character, secondary form) kI->k + ... (not a valid
Consonant to Consonant	k+k->kk (ক + ক→ক্ক) k+l- kl(ক + ল→ক্ল)	kallol- k + l+ l+o+l /(কল্লোল→ক +ল + ল + ** +ল (not valid) Note: ** symbolic representation of vowel

2.4 Dataset Preparation

Although Indic TTS corpus (developed by Indian Language Technology Research Center (ILTRC)) is available online with several Indian Languages including Assamese, it could not meet the requirement due to its limited size. A self-styled new dataset has been developed by these authors for experimental purposes collecting data from various domain specific sources. This dataset consists of 40,000 audio clips of different Assamese graphemes (vowel, consonant, compound character, features, words etc) in standalone machine. Performance is measured based on the result set observed with this standalone dataset. As mentioned earlier, AI-enabled TTS systems require large amounts of high-quality training data with pairs of text and corresponding speech recordings, our data set is very small as required for which variations are observed in the output. During the training process, the neural network learns to capture the complex relationships between text and speech features, such as phonetics, prosody, and intonation patterns.

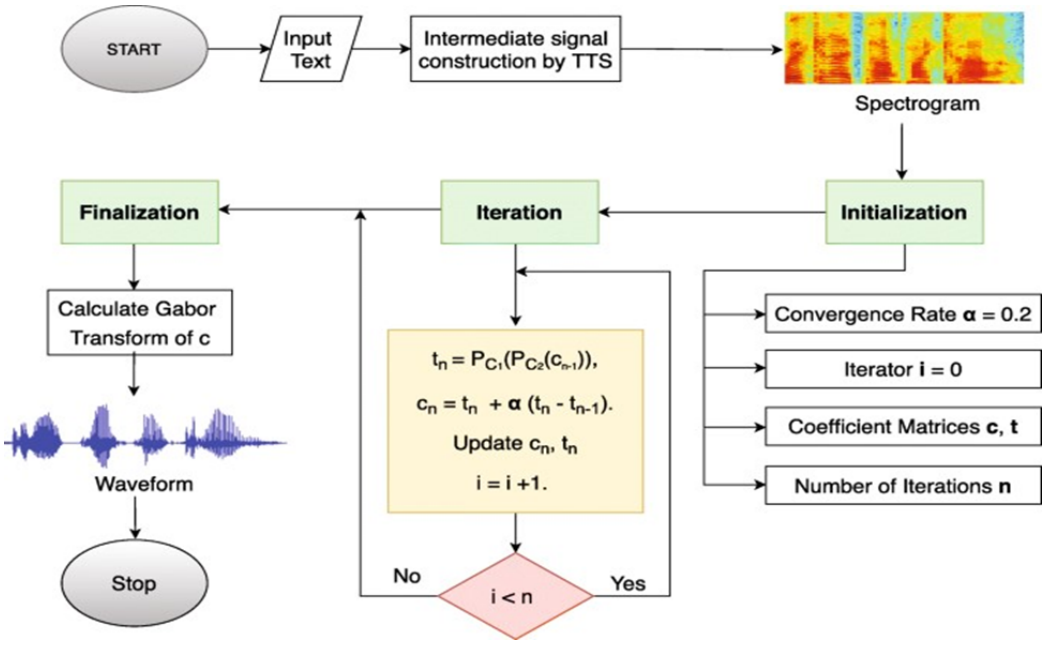

Fig 4. GLA algorithm

In this experiment, Griffin-Lim algorithm (GLA), Figure 4 is used for phase reconstruction in audio signal processing, particularly in the context of audio waveform synthesis from magnitude spectrograms. This process is iterative. In this work, a simplified Python implementation of the Griffin-Lim algorithm has been used for reconstructing an audio waveform from its magnitude spectrogram as suggested by[9].

GLA follows the following four steps for Audio generation:

Step 1: Initialization

i. For each time-frequency bin in the magnitude spectrogram of the input audio, start with a randomly initialised phase estimate.

ii. Utilising the initialised phase estimate and the magnitude spectrogram, compute the complex-valued spectrogram.

Step 2: Iteration Refinement

Follow these procedures again until you reach convergence or a predefined number of iterations.

i. Inverse Fourier transform: To estimate the signal in the time domain, compute the complex spectrogram's inverse Fourier transform.

ii. Magnitude spectrogram calculation

iii. Phase update

iv. Complex-valued spectrogram reconstruction

Step 3: Convergence Check

After examining the convergence criteria, it finds the difference between successive waveform estimates or the change in spectrogram magnitude and stops iterating if the criteria given are satisfied.

Step 4: Output

Output the final reconstructed waveform when convergence is reached, or the maximum number of iterations is reached.

3 Results and Discussion

In order to synthesize voice directly from text, we used Tacotron2 neural network architecture[10]. This produced a corresponding sequence of spectrograms, which were subsequently transformed into audio waveforms using a vocoder such as WaveNet of the Griffin-Lim technique. It is composed of two parts: a modified version of WaveNet that creates time-domain waveform samples conditioned on the predicted mel spectrogram frames, and a recurrent sequence-to-sequence feature prediction network with attention that predicts a sequence of mel spectrogram frames from an input character sequence. These models could be trained on large datasets of text and corresponding speech recordings to generate high-quality synthetic speech.

Several factors, like data size, learning rate, batch size, number of epochs, etc., are taken into account when training this model. Accuracy is the metric used to assess training and test model performance in machine learning algorithms. The examination of additional characteristics, such as recall rate (defined in equation 3 as properly predicted value to total prediction value) and accuracy (defined in equation 1 as number of correctly predicted values to total number of predictions), is used to determine performance.

The F1-Score in equation 4 is the ratio between average mean recall and precision.

Total size of Dataset: 40000 (approx) (Self-created)

Target language: Assamese

Experimental dataset: Five categories from different domain

$$Accuracy\ X = \frac{(Tp+Tn)}{(Tp+Tn+Fp+Fn)}$$

$$Recall\ R = \frac{Tp}{Tp+Fn}$$

$$Precision = \frac{Tp}{Tp+Fp}$$

$$F\ Score\ F1 = \frac{2*Precession*Recall}{Precession+Recall}$$

Where, Tp- True positive, Tn- True Negative, Fp- False positive, and Fn- False negative.

Five different datasets have been chosen from different sources as Set1 (collected from newspaper, 10k words), Set2 (collected from Scientific article, 8k words), Set3 (collected from E-journal), Set4 (collected from Books), Set5 (collected from Magazine). The parametric analyses of experimental results are shown in Table 3. With these five datasets, out of 40000 records, 36000 (approx) has been utilized for acoustic generation which 83 pc accuracy level has been maintained irrespective of data size. In this experiment, with earlier development of these authors (CNN model), the system was tested with collected text from three different Unicode languages and compared its performance with English language keeping the size of database as standard. The comparative results are shown in Table 4. The observed discrepancies in accuracy can be attributed to the greater inflectional nature of Indian languages. Second, the dictionary's size is yet another crucial consideration. As would be expected, greater population size leads to improved accuracy.

Table 3. Parametric analysis of experimental results

Size of Dataset	TP	TN	FP	FN	Accuracy (%)	Recall	Precision	F Score
9950	7000	1100	900	880	81.98	0.89	0.89	0.89
8300	6000	900	770	560	83.84	0.91	0.89	0.90
6700	4600	800	480	400	85.99	0.92	0.91	0.91
6120	4300	660	350	490	85.52	0.9	0.92	0.91
4700	3000	700	160	450	85.85	0.87	0.95	0.91

Table 4. Parametric analysis of experimental results

Target Language	Volume of database	Sample size (input)	Correctly Spelled	Accuracy in %
English	412	88	66	70%
Hindi	334	56	18	32%
Assamese	450	197	84	42%
Bengali	243	68	19	29%

Accuracy comparison of correctly spelled word searching in the sound database with single word input in ASCII vs Unicode languages

Next, using the CNN method, we evaluated our system using the same database with input as a plain text file that had roughly the same word count. Table 5 shows the second succession's performance. Finally, based on their performances, it has been discovered that the Assamese language accuracy level displays somewhat higher with G2P in comparison to the current CNN method as shown Table 6.

As given in Table 6 with three categories of data sets from Assamese language, it has been observed gradual improvement in accuracy among these four approaches where G2P claimed up to 89% accuracy which is higher than earlier development by Sharma and Kumar during 2020. However, this is the first endeavor for Assamese language (a regional 16-bit language with rich ligatures) where words are spelled based on their syllable using graphemes.

Table 5. Percentage of Accuracy experimented with CNN model

Target Language	Sample input size	CorrectlySpelled	Accuracy in %
English	112	62	56%
Hindi	140	55	40%
Assamese	163	58	36%
Bengali	98	31	32%

Table 6. Comparison chart of different approaches

Input size	Dictionary	HMM	CNN	G2P
Data Set A(100)	40	60	70	80
Data Set B(300)	50	66	83	86
Data Set C(1000)	60	70	85	89

4 Conclusion

The first step towards developing NLP in Unicode languages is the development of AI-enabled TTS for regional languages like Assamese. TTS conversion in Assamese language is found very rare in public platforms. It can be difficult to develop TTS for digital humans, especially when it comes to producing speech that sounds authentic and natural depending on the language and place. This is due to the possibility that speech produced by TTS systems built using conventional and statistical methods will sound robotic or mechanical, which consumers may find offensive. Furthermore, producing flexible and adaptive speech is necessary for digital human applications. This can be difficult because TTS systems depend on various elements, including datasets and the kinds of models and modules that are employed. This model will be further developed considering other parameters like male voice, female voice, sound quality, speech duration, pits intensity, impression etc. The dataset used for training purposes is considerably very small for machine learning and has to be increased three times the existing one.

In our first attempt, only manually recorded female voices were accumulated to produce an audio file with 40k sample sounds of different graphemes, ligatures, and some words. We compared this with three other languages (16-bit format) and English. Due to the structural similarity of this target language with other languages, there is scope to enhance the quality and performance with a fully AI-enabled system. The key challenges will include generating pronunciations for similar words and the addition or deletion of ending graphemes. While reading, a word ending with a vowel may be pronounced differently, resulting in different meanings. These problems must be addressed in the next development phase with larger ML tools.

Acknowledgements

Authors are thankful to the publishers for giving us the opportunity to share our research work. We are grateful to those without whose support we wouldn't have concretized the new concepts in this field. Not only that, the reviewers' comments also help us in a great manner. We always remember their valuable suggestions. Our sincere thanks go to all colleagues of the Department of Computer Science of USTM for their continuous inspiration.Authors are thankful to the publishers for giving us the opportunity to share our research work. We are grateful to those without whose support, we wouldn't have concretized the new concepts in this field. Not only that, reviewer's comments also help us a great manner. We always remember their valuable suggestions. Our sincere thanks go to all colleagues of the Department of Computer Science of USTM for their continuous inspiration.

References

1) Sharma A, Kumar P, Maddukuri V, Madamshetti M, Kishore KG, Kavuru SSS, et al. Fast Griffin Lim based waveform generation strategy for text-to-speech synthesis. *Multimedia Tools and Applications*. 2020;79:30205–30233. Available from: https://doi.org/10.1007/s11042-020-09321-7.
2) Pradhan A, Yajnik A. Parts-of-speech tagging of Nepali texts with Bidirectional LSTM, Conditional Random Fields and HMM. *Multimedia Tools and Applications*. 2023;83:9893–9909. Available from: https://link.springer.com/article/10.1007/s11042-023-15679-1.
3) Deka A, Sarma P, K S, Prasanna SRM. Development of Assamese text-to-speech system using deep neural network. In: Conference: Twenty-fifth National Conference on Communications (NCC). IEEE. 2019. Available from: https://www.researchgate.net/publication/331257172_Development_of_Assamese_Text-to-speech_System_using_Deep_Neural_Network.
4) Yolchuyeva S, Nemeth G, Gyires-Tóth B. Grapheme- to-Phoneme conversion with Convolutional Neural Networks . *Applied Sciences*. 2019;9(6). Available from: https://doi.org/10.3390/app9061143.
5) Khanam F, Munmun FA, Ritu NA, KSaha A, Mridha MF. Text to Speech Synthesis: A Systematic Review, Deep Learning Based Architecture and Future Research Direction. *Journal of Advances in Information Technology*. 2022;13(5):398–412. Available from: https://www.jait.us/uploadfile/2022/0831/20220831054604906.pdf.
6) Barman AK, Sarmah J, Sarma SK. Development of Assamese Rule based Stemmer using WordNet. In: Vossen P, Fellbaum C, et al., editors. Proceedings of the Tenth Global Wordnet conference. 2019;p. 135–139. Available from: https://aclanthology.org/2019.gwc-1.17.pdf.
7) Ahmad A, Selim MR, Iqbal MZ, Rahman MS. An encoder-decoder based grapheme-to-phoneme converter for Bangla speech synthesis. *The Acoustical Society of Japan*. 2019;40(6):374–381. Available from: https://www.researchgate.net/publication/336966498_An_encoder-decoder_based_grapheme-to-phoneme_converter_for_Bangla_speech_synthesis.
8) Sarmah J, Sarma SK, Barman AK. Development of Assamese Rule based Stemmer UsingWordNet. In: Proceedings of the 10th Global Wordnet Conference. 2019;p. 135–139. Available from: https://aclanthology.org/2019.gwc-1.17.
9) Liu H, Baoueb T, Fontaine M, Roux JL, Richard G. GLA-Grad: A Griffin-Lim Extended Waveform Generation Diffusion Model. In: IEEE International Conference on Acoustics, Speech and Signal Processing. 2024. Available from: https://arxiv.org/abs/2402.15516.
10) Mutawa AM. An end-to-end Tacotron model versus pre trained Tacotron model for Arabic text-to-speech synthesis. *Journal of Engineering Research*. Available from: https://doi.org/10.1016/j.jer.2023.08.016.