Real Time Closed Captioning for Bengali Multimedia

by

Avishek Paul
21301171
Mohammad Latif Mozammel
21201139
Prachurja Bhattacharjee
22101485

A thesis submitted to the Department of Computer Science and Engineering
in partial fulfillment of the requirements for the degree of
B.Sc. in Computer Science

Department of Computer Science and Engineering
Brac University
June 2025

Declaration

It is hereby declared that

1. The thesis submitted is my/our own original work while completing degree at Brac University.
2. The thesis does not contain material previously published or written by a third party, except where this is appropriately cited through full and accurate referencing.
3. The thesis does not contain material which has been accepted, or submitted, for any other degree or diploma at a university or other institution.
4. We have acknowledged all main sources of help.

Student's Full Name & Signature:

Avishek Paul
21301171

Mohammad Latif Mozammel
21201139

Prachurja Bhattacharjee
22101485

Approval

The thesis/project titled "Real Time Closed Captioning for Bengali Multimedia" submitted by

1. Avishek Paul (21301171)
2. Mohammad Latif Mozammel (21201139)
3. Prachurja Bhattacharjee (22101485)

Of Spring, 2025 has been accepted as satisfactory in partial fulfillment of the requirement for the degree of B.Sc. in Computer Science on June 20, 2025

Examining Committee:

Supervisor:
(Member)

Dr. Farig Yousuf Sadeque
Associate Professor
Department of Computer Science and Engineering
Brac University

Thesis Coordinator:
(Member)

Dr. Md. Golam Rabiul Alam
Professor
Department of Computer Science and Engineering
Brac University

Head of Department:
(Chair)

Sadia Hamid Kazi, PhD
Chairperson and Associate Professor
Department of Computer Science and Engineering
Brac University

Abstract

The widespread use of Bengali multimedia content on different platforms increases the need for an accessibility tool to cater to the Bengali content audience. Despite having tools for closed captioning pre recorded videos, there is still a need for a real time captioner to address the problem of captioning live programs. Closed captioning of live programs can make these contents more accessible to a wide range of people, including hard of hearing people and those who are not accustomed to the Bengali language. Due to environmental noise interference, temporal synchronization requirements, and overlap artifacts in streaming transcription systems, real-time closed captioning for Bengali multimedia content faces critical challenges. This paper presents a comprehensive framework for robust Bengali closed captioning that integrates a novel architecture optimized for multimedia applications. Our approach combines VAD-enhanced audio preprocessing, Silero-based filtering with Bengali-optimized thresholds (speech probability 0.25, ratio 0.15), and adaptive spectral noise suppression using conservative subtraction (=1.5) with 30% spectral floor protection to preserve conjunct consonants and tonal variations critical for Bengali intelligibility. To establish optimal acoustic model foundation, we systematically evaluated custom -xls-r-300m and whiper-small fine-tuning on 20 hours of Mozilla CommonVoice data versus pre-trained alternatives, demonstrating that larger training datasets (300+ hours) provide 35-45% superior performance for multimedia captioning scenarios. Our overlap prevention framework employs sliding window deduplication and cross-stride output comparison with Bengali morphological fuzzy matching. While it filters many overlap, there is room for improvement in overlapping prevention. The system maintains sub-second latency with minimal computational overhead, advancing accessible Bengali multimedia consumption for hearing-impaired communities and multilingual audiences. However, further work is needed on handling real-life multimedia challenges including speaker variations, tonal variation due to frequent emotional changes, background music interference, and cross-talk scenarios to achieve broadcast-quality transcription accuracy for diverse multimedia content.

Keywords: Deep Learning, Transformer, LLM, Supervised Learning, Speech Recognition, BERT, Speech to Text, Natural Language Processing (NLP), Fine-tuned, Streaming ASR, Real Time, Closed Captioning, Subtitle, Multi-modal, Wav2vec2, Whisper, N-gram.

Table of Contents

Chapter 1

Introduction

Automated speech recognition systems are seen on a large scale in today's world. A technology that once reverberated across the world is now one click away to anyone owning a smart device. ASR systems have seen a great deal of progress in recent years, and that includes the ability to give real time feedback like the captioning system that youtube uses for their videos. However, it has to be said that such real time captioning systems are reliable only if there is ample labeled speech data to train these systems. Bengali is one such language that has not seen a lot of work done on building a real time closed captioning system for videos. It is quite underwhelming to see that despite Bengali being one of the top ten languages spoken in the world, there appears to be a lack of research on Bengali in NLP. It could be for a variety of reasons: most applications have an English user interface and so researchers do not feel the need to invest time and resources working with a language that does not see a lot of light in technological systems. As mentioned earlier, machine learning systems, particularly speech recognition, require copious amounts of data to function satisfactorily.Bengali as a language has many intricacies and quirks in grammar, pronunciation and dialects which can prove to be a difficult task for researchers and scientists to generalize for NLP tasks. Further, data related to Bengali is scarce compared to languages like English or German, and this can act as an obstacle for any work in NLP. In this paper, we will demonstrate real time bengali closed captioning using popular deep learning models such as OpenAI's whisper small and FacebookAI's(Now MetaAI) wav2vec2 xlsr 300m models along with some of their fine-tuned versions. We will also take the help of some audio preprocessing techniques such as noise handling, dynamic stride creation, two context handling and overlap handling to ensure the audio is prepared for real time transcription of the aforementioned models.

1.1 Research Statement

In this thesis our aim is to create a system that allows us to generate real time captions for Bengali multimedia content. This will help the hard of hearing people to understand the content better. Moreover, it will also make Bengali multimedia content accessible to people who aren't fluent in Bengali. Therefore, this study aims to make real time Bengali closed captioning system to make Bengali multimedia accessible to more people.

1.2 Research Objectives

- Train a model that performs well for low resource languages and has low latency to implement real time transcription
- Using existing dataset for model train and validation.
- Build a dataset with audio, video and text data to test model
- Use NLP and ASR techniques for preprocessing and postprocessing.

Chapter 2

Related Work

2.1 Survey Methodology

For this research, we had to undergo extensive literature review, especially on streaming ASR techniques and on low resource language processing.We have gone through low latency models which performs well in real time. Moreover, works on multimodality has also reviewed by us to understand working with video, audio and text data together. We made sure to pick out journal articles and conference papers that are recently published so that we can understand how modern natural language processing techniques and deep learning methods are implemented. Further, we tried to pick out highly cited papers or papers that are published in well-knowned conference or Q1/Q2 journal.

2.2 Related Works

Mach´a˘cek et al. (2023) [15] introduced real time transcription to OpenAI's invention- the Whisper automatic speech recognition system. Using their own ideas along with the findings of other researchers, they have attempted to implement a version of the whisper model which can showcase transcription on the go with an acceptable amount of delay. The authors of the paper used the large-v2 model of Whisper due to it achieving the highest quality results of all the other sizes. The primary idea is to store the audio in an audio buffer and use whisper to process the data. Afterwards, an algorithm called LocalAgreement2(Liu et al.,2020) is used to implement the simultaneous streaming mode feature.The authors further talk of a functionality called the " Update Loop" which controls how the chunks of audio captured would be processed for real time transcription. A parameter " MinChunkSize" controls the duration of audio processed per update cycle and this parameter can affect both latency and quality of the transcription. Since the LocalAgreement algorithm outputs the longest common prefix on n consecutive source chunks, this, along with the fact that whisper adds word level timestamps, can be used to avoid repetition in the transcription.The authors have used the dev set of the ESIC corpus(Machácek et al.,2021) for implementing English, German and Czech ASR. The corpus consists of 179 documents. It contains audio files with human written transcriptions and time markers at a word level(Dominik et al.,2023). For quantifying how well audio was being transcripted into real time text, they used WER as a metric. The authors measured the difference between the timestamp when a word

appears in the actual transcript and the timestamp when Whisper Streaming outputs the same word. With voice activity detection(VAD) on, %WER is observed to be lesser for german than if VAD is on. However, for the english language, VAD makes little difference in %WER. Latency wise, the figures look close to one another but both VAD on and off. For English, with a MinChunkSize of 1 second, the average latency was observed to be 3.3 seconds while for German and Czech, it was 4.4s and 4.8s respectively. Furthermore, if larger chunk sizes are taken, it results in a higher quality transcription but a longer latency too.

Moritz et al. (2019) [3] proposed a Triggered Attention end-to-end speech recognition model to improve the recognition of streaming speech. Despite working well in mapping speech input to text output, attention based models face limitations in streaming speech recognition. On the other hand, CTC works well in frame-level alignment but accuracy is lesser than attention based models. Hence, the authors introduced the Triggered Attention architecture where a CTC based trigger network controls the activation of an attention based decoder. The encoder of this model architecture is based on a CNN with VGG structure, followed by BLSTM neural network. The model is trained with a multi objective loss function, combined with CTC loss and attention loss. Hence, the model can both grasp the frame level alignment of CTC and optimize the more powerful attention model. The authors used three ASR datasets for evaluating the model, including 80 hours of English training data from Wall Street Journal, 174 hours of training data from HKUST (Mandarin telephone speech dataset) and 960 hours of English training data from LibriSpeech. English and Mandarin data were chosen to generalize the model for multiple languages and different sizes of training data. Across both languages, the Triggered Attention model consistently outperformed the baseline full-sequence attention model. Using location-aware attention the TA model got a WER of 16.2% on the test set where the full-sequence attention model got 18.6%. For the HKUST dataset, in comparison to 33.4% for the baseline, the TA model got better performance with a test set CER of 31.9%. Finally, for the LibriSpeech TA model achieved a WER of 7.4% on the 'clean' test set but the baseline got 6.1%. In contrast, better performance was seen in the 'other' data set where TA achieved 19.2% and baseline got 20.0%. A delay of just 80 milliseconds was achieved by the model with the optimization of the look-ahead parameter to balance accuracy and latency. These results indicate that the Triggered Attention model not only improves recognition accuracy but also enables streaming recognition with minimal delay.

Martin et al. (2021) [8] analyzed automatic alignment of subtitles and transcription in this paper. The Authors introduced a deep sync tool (pertaining multilingual representation language model) which can solve various existing problems for example-synchronization, subtitles that do not match with the audio, and censorship. Importantly, Re-speaking issues in live shows on television where speaker's hearing and lipsing coordination causes delay is also mentioned in this paper. In this paper, The main datasets consist of different types of videos and according to their model build up, they tested the models with those datasets . Pretrained BERT fine tuned with transformer learning is used for word vector representation. Among many versions of BERT, DistilBERT is used for being a faster and compact model which also retains it's ability to understand language. The tool Deep Sync performs well with

the help of ASR (which generates transcription) against delay. In terms of similarity checking, an alignment scheme based on time is performed if no similarity is found. For word shifting and adjustment, the Needlaman-Wunsch algorithm is used. Transcription based alignment is used for creating transcription search matrices to describe word sub sequence which are the probable candidate for alignment. For time-based alignment, the algorithm employed is the Inertia algorithm ,in which the time is represented by certain subtitles and brought forward representing the delay. In case of the failure of the inertia algorithm , this paper talked about Queued subtitles realignment. The results of the paper is appreciable as demo video 1, aligned at start time was 70.06% and aligned at end time was 69.76% without alignment with transcription and it became aligned 94.21% during starting time and 88.95% during ending time which is much better performance. But with the same subtitle when aligned with time , the performance is worse than before getting aligned with start time is 32.67% and with end time is 39.60%. In the rest of the videos exactly the same things occur where subtitles aligned with transcription are giving better results and subtitles aligned with time are giving results which are not up to the mark.

Carrasco et al. (2019) [1] introduced the Sub-Sync framework in this paper which automatically synchronizes audio visual contents with it's subtitles. In this paper, Sub-Sync framework is applied on live broadcast, live scripted programs, pre recorded programs and mixed programs. For this, the datasets used by authors are two parts of the TV magazine-programs of different channels and live broadcast of a football match. For ensuring synchronization, this paper mentioned three stages. First stage is transcription ; in which the ASR engine has been used for continuous word flow and corresponding words with audio-visual audio and to store in transcription storage. Second stage is chronization which itself comprises two stages. Among them, the first stage compares words stored in transcription storage with subtitles and in the second stage, with the help of the Needleman-Wunsch algorithm similarities between them are being processed for best allignment. Incase of no similarities between words stored in transcription storage with subtitles, Inertia-algorithm is used. In the third stage, the subtitle transcriptions and the video with audio media are synchronized. In this paper, the ASR used is google. Google API allows inclusion of context words also. The Levenshtein algorithm is used by the authors to calculate word distance. After running their models for different datasets, they get for re-speaking, the average delay is 9771 sec and Standard Deviation (SD) is 2552 sec. On the other hand, manual synchronization is showing an average delay of 453sec and SD 1974 sec . In the case of the Needle-Wunsch algorithm, aligned output is showing an average delay of 303 sec and SD is 988 sec. Similarly, the Inertia algorithm is showing an average delay of 710 sec and 593 sec. By analyzing the results, the authors come up with the decision that both of the algorithms are showing better results than respeaking and are almost similar to manual synchronization.

Tündik et al. (2020) [7] introduced a punctuation model, based on RNN with low latency, aiming to improve closed captioning to apply it in real time. In this paper, the authors used English and Hungarian datasets to evaluate this model, mainly DHH users watching broadcast media. in this paper the performance of the proposed model is compared with both traditional methods (e.g. MaxEnt) and models which are considered benchmark in real time application. For tolerating maximum

latency, the authors used ASR decoder which provided an approximate word by word transcription in real time. For RNN models, the data is splitted into constant length and smaller scale by the authors to ensure no overlap and no skip. For most common words, a vocabulary is created with k+1 entries and for unknown words, using it a common vector is utilized for mapping the rare words. On basis of pretrained embeddings weight matrix for embeddings was also introduced. In this paper, authors build up RNN models from LSTM cells. Connecting LSTM for data flow can be more powerful if BiLSTM is introduced according to the authors. Systematic grid-search optimization was also performed on the validation set for hyperparameters. Finally, for each hyperparameter's concluding values of English and Hungarian datasets the authors used WE-LSTM and WE-BiLSTM. In their models, they used CRF on top of RNN to get help in Named Entity Recognition (NER) . Contrasting with modern models , the authors used vanilla LSTM models to solve the punctuation problem in an on-line closed captioning use case. For punctuation mark Comma, The result of their model for Hungarian manual transcripts is, in offline mode, WE-BILSRTM's F-1 score is 71.2. For WE-LSTM-CRF , F1 score is 72.3. Both of the results are better than the F1 score of MaxEnt which is 65.5 . Again for on-line mode, F1 score for WE-BILSTM and WE-LSTM-CRF are 71.1 and 70.4 respectively. But MaxEnt's F1 score is 64.2. The authors are getting best results from the WE-LSTM . This model is overperforming the other models.

Samin et al. (2024) [17] discuss the challenges faced by Automatic Speech Recognition (ASR) systems specially in adapting different domains or environments in this paper. They also address the problem as a domain shifting problem. In this research article, the authors aimed to improve robustness and generalization of ASR systems across multiple domains , particularly for Bangla language. The authors used BanSpeech (a new multi-domain Bangla ASR evaluation benchmark) . The dataset contains 6.52 hours of speech from 13 different domains, most of them are spontaneous speech collected from youtube and also transcribed from human annotations. Besides, they used SUBAK.KO dataset on diverse domains to see the performance. In this research article, the authors used CNN-based models for a higher number of MFCC's . They downloaded broadcast speech as waveform audio file (WAV) and changed the audio file name. Then did preprocessing by removing commas, brackets replace spaces with underscores. They converted the audio file to a bitrate of 256 kbps and 16 KHz mono channel WAV file. Those audio files were automatically divided into smaller chunks and the initial length was too big according to the researchers. So, they used a threshold of - 40 decibels relative to full scale (dBFS) and 0.2 s as min length. They kept audio file range (0.7 - 35) sec . For the annotation process, three stages have been mentioned. In the first stage, they used Google speech to text (STT) system for transcriptive native speeches. After manual correction, transcriptions are sorted in plain text file format (.txt) Coming back to the CNN model, the authors used three types of features from the frequency domain which are mel- frequency cepstral coefficients (MFCC) , mel-frequency spectral coefficients (MFSC) and power spectrum. According to the research, frequency domains exhibit robustness to variations, lead to substantial; increase in space dimensionality, and enhance computational efficiency. In this research paper, to get MFCC feature, raw waveforms go through a pre-emphasis filter , then computed in a short time Fourier Transform (STFT) , lastly discrete cosine transform is

applied. While training, the authors used Connectionist temp[oral classification loss function. Finally after testing, the authors get WER testing wav2vec XLS-R is 40.62 and CER 14.02 for BanSpeech. For the SUBAK.ko test , WER is 7.56 anD CER is 2.02. On the other hand, for the CNN model , for banSpeech , WER and CER are respectively 67.65 and 30.83. For SUBAK.KO WER is 13.94 and CER is 4.02. Lastly for Whisper, BanSpeech got WER and CER 55.92c and 15.94 respectively and for SUBAK.KO, WER and CER is 22.15 and 4.96 respectively. So, the authors found self supervised wav2vec is performing better than other models.

Fathullah et al. (2023) [14] extended the capabilities of LLM by direct attachment of audio encoder to perform speech recognition in this paper. The authors also conducted experiments to see if the LLM could be kept unchanged during training, scaling up the audio encoder and increasing the encoder striding to produce fewer embeddings . For the dataset, the authors use multilingual LibriSpeech (MLS) which has a 50k hours ASR corpus derived from read audiobooks on Librivox. Additionally it consists of 8 languages. Among them, the dataset is predominately in English with 44.5k hours but Portuguese and Polish have only 161 hours and 103 hours respectively. For solving this imbalance, the authors used a strategy of oversampling. For testing their research , the authors used the smallest LLaMA-7B model. The self attention parameter of this model can be adapted using parameter efficient Low-rank Adaption (LoRA) , keeping all the other parameters frozen. Again, the authors also investigate the importance of LLM by replacing it with LLaMA with various BLOOM models. While analyzing the models, the authors also introduced ASR-LLM problem which can be possibly reinterpreted as a translation task where LLM needs to regurgitate the information in the audio sequence. About the audio - encoder setup, 80-d filter bank feature with 10ms frame rate is set including convolutional feature extractor stride of 8 followed by linear layer. They mentioned about 512 dimensions and 18 layers of non-macaron Conformer blocks. For pretraining the encoder, a final layer is used using a CTC loss with vocabulary size of 1547. According to the authors, approximately 72 million parameters could be significantly improved by scaling size up. Coming back to Low-Rank Adaptation (LoRA) approach, default hyperparameters are set to rank of R is 8, and alpha is 19. Finally, they trained their model with Adam optimizer setting Beta1 as 0.9 and Beta2 as 0.98 along with learning rate 20k. After training the models, the authors find their LLaMA (decoder only) model outperformed the system with lower rate along with monolingual models by 18% and 10% on average word error rate.

Priya et al. (2021) [11] proposed to develop a novel deep sequence model-based ASR system with improved spell correctors in this paper. For the dataset, the authors used monolingual speech data that is annotated with text levels spoken by native speakers and their speech by native speakers and their speech is collected as scripts for seven Indian languages. They checked an average of 3500 sentences which were spoken by Ind-Aryan and Dravidian languages except Marathi. Besides, Indian script code for Information Interchange (ISCII) is used for benchmark reference for collecting the scripts. Considering the probable challenges of speech to text models and for utmost accuracy, the authors used RNN-GRU and Indic - BERT models for their proprocessed work. The authors inspected every instance of speech overtime by windowing and framing. Input speech of their dataset is framed by a window

size of 20 to 25 ms with 10ms stride. The application of discrete cosine transform (DCT) on input speech frames is done for attaining MFCC. In their construction of deep sequence recurrent blocks, a 1D convolutional layer is used along with seven GRU layers and 250 cells each. The designed GRU by folding it into two gated units such as update and reset gate. The authors used Adam optimizer to optimize their model. For better prediction of unseen data by the models, they incorporated the values such as L2 regularization of 1e-6, momentum values as 0.9 , 20% dropout , learning rate 1e-3 and then they trained the model with 50 epochs. Later on, for removing spelling errors of the output of RNN-GRU model, they used pretrained Indic-BERT language model on 12 Indian languages. Spell error detection was calculated by Levenshtein algorithm (lev(a,p)) which mainly calculates the difference between the actual a and predicted p tokens and do necessary insertion, exclusion and substitution to match the equation. According to the output of the models, the authors got good performance for GRU in which for Bengali language , validation was 0.80. On the other hand, for GRU+Indic BERT , the validation is 0.72. They come up with the conclusion that , because of the large number of morphological variations , generalization error has increased which can be solved by leveraging the size of high quality Indian speech data.

Jian et al.(2023) [16] proposes a decoder-only architecture for converting speech to text. While the paper maintains that it is a decoder-only structure, a acoustic encoder was used to convert speech into embeddings which will then be received by a LLM decoder. Before the encoding process, a CTC compressor is used to truncate the input sequences in order to ensure the input audio and the text associated with it are in sync with one another. The authors experimented with two types of CTC compressor: a "blank removal" process where all frames that predicted the blank symbol are removed and "frame averaging" process where the concealed states of back to back frames is averaged after all their CTC predictions belong to the same class. The difference between a traditional encoder-decoder and this proposed architecture is that the audio encoder mentioned earlier is pre trained to receive audio signals whereas in an end-to-end system, the encoder needs to be trained before the audio is passed in for generating embeddings. The system implements instruct learning where an instruction is appended to the start of every training example in order to guide the model during the training and validation. LoRA is implemented to four attention matrices in each layer in order for the training process to be more efficient due to dealing with fewer parameters and to be less computationally heavy. To really test the strengths of the decoder-only architecture, from-scratch training is used where the text prompt, audio encoder and the compressor is replaced with a randomly initialized 2D convolutional encoder. The authors used 13 languages, mostly european, and the training data is mentioned as anonymous in the paper. For testing, CoVoST 2 dataset was used. CTC loss along with cross entropy was used for optimization. Comparing to the seq2seq model which was set as benchmark by the authors, the LLaMa performed better with a BLEU score higher than 0.9.

Chen et al. (2023) [13] introduced VAST, a Vision Audio-Subtitle-Text omni-modality foundation model and dataset which is used to perceive and process four modalities to improve the tasks, including retrieval, captioning and question answering. In order to do so, they first created a dataset VAST-27M to address

the absence of omni-modal captions. VAST-27M contains 27 millions video clips from the HD_VILLA_100M dataset which are automatically captioned using three modalities, including video, audio, and subtitle. The authors suggested a two stage automatic pipeline to automate the captioning. Firstly, a video captioner and an audio captioner trained on publicly available video and audio caption corpora were used to generate single modality captions. Then, the audio and video captions along with subtitles were merged using Vicuna-13b, an off shelf LLM, to create the omni-modal captions for the VAST-27M dataset. This dataset is used to pretrain VAST, a fully end-to-end transformer architecture, consisting of a audio encoder, a video encoder and a text encoder. Beats was used as an audio encoder, BERT as a text encoder, and ViT as a video encoder. The audio encoder is used as both single-modality encoding and omni-modality encoding/decoding using cross attention layers. The model is trained using three pre-training objectives following Omni-Modality Video-Caption Contrastive Loss (OM-VCC), Omni-Modality Video-Caption Matching Loss (OM-VCM), Omni-Modality Video Caption Generation Loss (OM-VCG). This architecture enables VAST to handle a wide range of tasks involving vision-text, audio-text, and multi-modal video-text. In audio-text benchmark, VAST achieved significantly improved five new state of art results due to the large VAST-27M dataset. It also surpasses all types of multi-modal video-text benchmarks particularly surpassing GIT2 foundation model by 2.1, 67.6, 5.0 and 8.0 CIDEr points on MSRVTT, VATEX and TVC captioning benchmarks. Due to the use of open-sourced cross-modality corpus in training the video and audio captioners, VAST-27M dataset and VAST model may include those dataset's and model's biases.

Chen et al. (2023) [12] proposed VALOR, a Vision-Audio-Language Omni-Perception model to address multimodal learning across vision, audio, and text. In previous vision-language models audio integration was absent which VALOR tackles efficiently. To overcome the unavailability of proper vision-audio-language dataset, the authors created VALOR-1M dataset for pretraining and VALOR-32K for evaluation of audiovisual-language tasks. The dataset is composed by choosing videos from AudioSet. VALOR-1M is created from the unbalanced dataset and VALOR-32K is created from the balanced dataset of AudioSet. Then the videos are annotated manually using three steps. In comparison to other video-language datasets, the VALOR dataset's ACD metric Is much bigger. Moreover, this dataset has stronger correlation among audio,vision and language than other datasets. VALOR model's architecture involves an audio encoder, a text encoder, a vision encoder and a multimodal encoder. As text encoder, BERT is used and two vision encoders, CLIP and VIdeoSwim transformer, are used. Furthermore, an Audio spectrum transformer is used as the audio encoder which is pretrained on AudioSet. The authors have used pre-trained BERT as the multimodal decoder. In order to pretrain the model, two tasks, Multimodal Grouping Alignment (MGA) and Multimodal Grouping Captioning (MGC), are proposed.The MGA task is designed for developing fine-grained alignment of text and other modalities and for the MGC task, casual masked language modeling are used. Valor produces better performance than previous models across different benchmarks. Across datasets like VATEX, MSRVTT and DiDeMo, Valor improves performance during retrieval by 3.8% and 12.7% respectively on video-text and audio-text retrieval task. Valor has established new state-of-the-art

results in audio visual captioning outperforming model that used only visual data. Strong results were also received in audio to text tasks gaining up to 38.9% on retrieval benchmarks like ClothoV1. The model's tri-modal approach leads to better generalization across tasks, showing that incorporating audio significantly boosts performance in tasks like audiovisual retrieval and captioning.

Shi et al. (2021) [9] introduced an acoustic model EMFORMER to improve low latency streaming speech recognition by optimizing a streaming transformer with augmented memory (AM-TRF). Emformer solves the problems of high computational complexity, latency and training inefficiency without sacrificing accuracy by optimizing memory handling and parallelization in models for low latency speech recognition. While building upon AM-TRF, EMFORMER brings the following changes for improvement: caching of key and value projections from previous segments to reduce the computational load for the left context, carrying over memory vectors from lower layers enabling parallelized training, removing the attention between the summary vector and the memory bank, mitigating context leaking by making careful adjustments to how look-ahead contexts are handled during training. The authors used EMFORMER for the encoder of a hybrid model and a transducer model. For the hybrid model, the context and positional dependent graphemes are output units. The standard Kaldi LibriSpeech recipe was used to train the system. They have also used 80-dimensional log Mel filter bank features and applied speed perturbation and SpecAugment during training. For the transducer model, byte pair encoded 1024 sentence pieces were the outputs. Pre-trained EMFORMER encoder from the hybrid system and two LSTM layers constructed the predictor. During beam search for shallow fusion, a neural network language model (NNLM) was used. Emformer achieved a WER of 2.50% on test-clean and 5.62% on test-other for medium latency of 960 ms. Compared to the AM-TRF baseline, it's a 17% relative WER reduction on test-clean and 9% on test-other. Moreover, Emformer achieved WERs of 3.01% on test-clean and 7.09% on test-other outperforming LSTM baselines in low latency of 80 ms. Due to its ability to parallelize block processing, EMFORMER displayed a 4.6-fold speedup in training compared to AM-TRF making it highly effective for real-time speech recognition tasks. EMFORMER has also achieved an 18% reduction in RTF. Moreover, The model has reached an RTF of 0.13 for medium latency scenarios in real-time decoding.

Kannan et al. (2019) [2] presented a E2E multilingual system for low-latency, real-time applications and addressed the challenges of creating a multilingual ASR system. They have focused on nine Indic languages to achieve their goal of developing a low latency ASR to handle real world data (unbalanced data) and improve accuracy across those languages even with limited data. The authors have covered a total of 37,000 hours of audio across those nine Indic languages. The dataset is divided into training and test sets for each language The languages vary significantly in the number of available data. With 16 million utterances, Hindi has the most data. On the other hand, Urdu has the least data with 443,000 utterances. Bengali has a moderate amount of data of 3.9 million utterances. Their proposed solution is based on a streaming version of RNN-T (Recurrent Neural Network Transducer). It consists of an encoder network, prediction network and joint network. Eight Long Short-Term Memory (LSTM) layers compose the encoder network. Each layer has

2,048 hidden units followed by a 640-dimensional projection layer. The prediction network is similar to the encoder's structure. The joint network which combines the output from both the encoder and prediction networks has 640 hidden units. In order to produce the final probability distribution over the next output symbol, the joint network uses a softmax layer which is shared across nine languages. Three methods are explored by the authors in order to address the data imbalance: one-hot encoded language vector incorporation, adding adapter modules and data sampling. The authors observed only using data sampling did not solve the imbalance data problem as it increased WER rate. However, incorporating language vectors and adapter modules produces the best outcome. There is a 10% relative reduction in WER across all languages compared to conventional systems using the multilingual RNN-T model with adapter modules and language vectors. Greater improvements were seen in smaller languages like Kannada and Urdu where WER reductions are 34% and 25% respectively.

Xiusong et al.(2020) [6] propose three methods to enhance the performance of acoustic models in a low resource environment. The first method they have proposed is a modified triphone network based on multitask learning. MAT with multitask learning refers to performing a "tri-task" where from all frames from a set of T training frames is mapped to a set of tri-labels and another task "Mono-task" which is similar to the previous task but set of labels is mono-labels in this case. Combined with a hyper parameter alpha, these two tasks help to optimize the parameters of the neural network. Using these two tasks together ensures that the model generalizes efficiently without overfitting. Another method that is suggestd is the soft one hot label(SOL) instead of just one hot encoding(since this may lead to overfitting). Soft one hot label assigns multiple probabilities to a phenome but assigns the highest probability to the true sound of the phoneme and assigns a smaller probability instead of zero to the other sounds which may be familiar to the phoneme. This will result in the neural network understanding the similarities between the sounds and therefore lead to better fitting. MAT and SOL are combined to form the final loss function which makes the model less rigid to fitting and prevents overfitting. Finally, the authors write about the technique of feature combination. Commonly used features in ASR models like Mel Frequency cepstral Coefficients(MFCC) and Filter Banks(Fbank)(Xiusong et al) are combined with Feature-space Maximum Likelihood Linear Regression(FMLLR) so that the ASR can distinguish between different speakers. This can result in more correct and in depth frames for the speech signals. The dataset used in the experiment is the THCHS-30 which consisted of around 10 hours of training data comprising of 3000 mandarin utterances. The language model, a 3-gram model, includes 48000 words. To summarize the results, all three models combined produce the lowest CER%(character error rate) if an LSTM- HMM architecture is used. For MAT training, for alpha = 0.9, optimal results were seen.

Emir et al.(2020) [4] claims that at the time the paper was being written, modern ASR systems were not able to automatically transcribe a singing voice with great accuracy. Therefore, the authors have proposed a variant of a neural network architecture that may help alleviate this issue. The proposed neural network architecture is a dilated convolutional neural network with self attention and will act as the acoustic model. It Will consist of six convolutional layers with 3x3 filters. To

reduce the size of the feature vectors, they have implemented max pooling with a factor of 2. The six layers will be followed by a TDNN-f(Time Delay Neural Network) since they are popular in ASR systems for successfully capturing temporal context. The TDNN consists of 9 hidden layers. Factorization is used in this network which will help to decrease how many parameters there are in the layers. Right before the output layer, a multi head time-restricted attention layer is introduced in order to prioritize which part of the temporal context to focus on. As a language model, the authors write about three different models, 3-gram and 4-gram models with a maximum entropy object built using SRILM and one built using a recurrent neural network. Under "experimental setup", the authors have mentioned a hybrid model where a GMM-HMM triphone model is first trained to produce phone-level alignments. Depending on this model, the available data is cleaned and using this data, the neural network architecture mentioned before is trained. Three regularization methods are used to avoid overfitting and dropout scheduling is used.For the dataset, the authors made use of the DAMP- Sing! Dataset by Smule. In the dataset contained monophonic karaoke recordings of pop songs with lyrics synchronized to timings amounting to almost 150 hours of trainable audio data. Looking at the results, a bigger dataset led to a reduction in the WER% as the models were able to generalize better. Further, The CTDNN architecture showed much better performance in terms of WER% compared to a GMM-HMM baseline.

Niko et al.(2020) [5] first mentions how Attention-based encoder-decoder architectures like the transformer model are excellent at offline ASR tasks but cannot be applied very easily for real time transcription ASR. So the authors propose a solution to this issue where they apply time-restricted self-attention to the encoder,and the TA(time restricted) concept to the encoder-decoder attention mechanism of the transformer model in order to make real time transcription possible. In order to implement the time restricted encoder, the encoder was built using a two layer CNN module called encCNN and E self attention layers. The self attention layers consist of a multi head self attention layer and two feed forward neural networks which are separated by a reLu activation function. The future context of the input sequence from the CNN module is kept at a fixed size to make sure that the latency of the encoder architecture can be controlled. This is what the authors maintained as time restricted self-attention. The decoder uses the triggered attention(TA) concept. The concept requires that the encoder state sequence and the label sequence align with each other so that the attention mechanism only focuses on previous encoder chunks along with a fixed number of future chunks. So this alignment is forcefully accomplished using a CTC(connectionist temporal classification) evaluation function in training the decoder model. Once the alignment is performed, the TA mechanism controls which parts of the encoder output to focus on. An algorithm for this partnered behavior between the triggered attention concept and CTC is also presented by Niko et al.. For the dataset, the authors have used the LibriSpeech data set that consists of a list of read-out-loud english audio books. The CTC-triggered attention algorithm is compared to other variants like CTC beam search and attention beam search and the authors have found their algorithm to have generated a lower WER than the other algorithms mentioned.Additionally, an overall delay of 2.2 seconds was observed.

Javier et al.(2022) [10] first discusses how it can be difficult to implement real time speech recognition functions in ASR models because of the fact that in streaming mode, the full audio is not available and hence, it is difficult to predict the current word while ensuring that it is coherent with the word that comes after it(as the future context may not be captured in a streaming mode system). Another issue that was addressed was about the optimality of default pruning techniques in the decoding process because of the unavailability of the entire audio data. In the paper, the authors discuss changes and adaptations to the BLSTM acoustic model and the interpolated neural language model together to make streaming as efficient as offline ASR. The authors mentioned a sliding window technique which was used to help the BLSTM model capture a small portion of future context in every chunk of audio. This way, just like in an offline asr mode, the BLSTM now can work with both past and future information to predict the current phonemes. As mentioned earlier, adaptations in pruning techniques were also introduced by limiting the size of the sliding window so that just enough future context is captured for an accurate prediction. Once the phonemes are predicted, they are passed into an interpolated neural language model which refines the phonemes to form words for transcription. The models were then evaluated on the LibriSpeech and TED-LIUM release 2 speech corpus. Audio tracks from video talks along with their STM files were used to calculate the WER of the model. Furthermore, the authors have used the RTVE2018 consisting of TV shows that are broadcasted in public spanish TV. Results have shown in the paper that increasing the window size up until the minimum chunk size led to a decrease in WER. Relatively larger window size of 1s led to a better performance when the RTVE set was used and the authors believe the TV shows consisted of "heterogeneous conditions" and therefore larger context windows led to better performance.

Chapter 3

Work Plan

We have three phases in our thesis. Pre-thesis 1, Pre-thesis 2 and Final Defense. In the begining of pre thesis 1, we built our team and approched our supervisor for topic selection. Later, by consulting with our respected supervisor, we selected our topic "Real Time Bangla Closed Captioning for Bengali Multimedia." After finalizing our topic, we started reading papers which are related to our topic. Later on, we started writing summaries of the reviewed papers and prepared our report for pre-thesis 1. The timeline of our pre-thesis 1 is 4 months from July to October. In pre-thesis 2, we are looking forward to completing our maximum works. Mainly , we will collect datasets of audio files and build our model. We will collect existing ASR datasets and use it to fine tune our model for Bengali. Mainly, we are checking if our model can generate correct transcription from bengali audio in pre thesis-2. After that, we will make our poster presentation and report for p2. We will get 4 months from November, 2024 to February, 2025. In our pre-thesis 3 , we will try to fine-tune our models more and apply techniques to make our model work for real time closed captioning. To test our model properly we will create a dataset with audio, video and text. We will also write our final report during pre thesis-3. After that, we are aiming to develop a website for integrating our work and publish it. Lastly, we will sit for our final defense.For final defense, we will get 4 months from March, 2025 to June, 2025. The entire working process of our thesis is represented in a Gantt Chart.

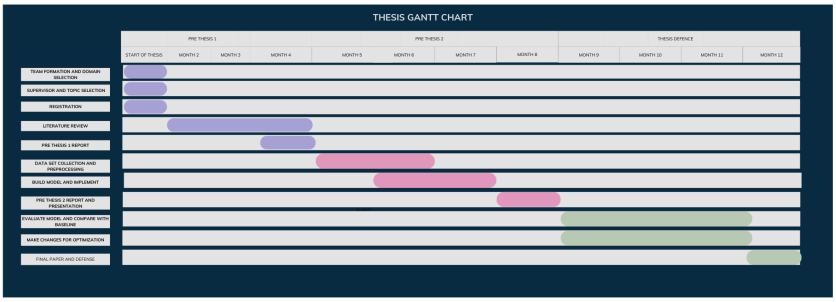

Figure 3.1: Thesis Workplan Gantt Chart

Chapter 4

Methodology

4.1 Training Methodology

4.1.1 Whisper-small

We have trained the whisper the model using the Mozilla Common Voice Bengali dataset loaded from the HuggingFace. The dataset is loaded in a streaming fashion for better memory handling. Once the dataset is loaded, it is preprocessed. The prepropcessing part involves turning the audio to Log-mel-spectogram, padding to meet the 30 second input limit of whisper.

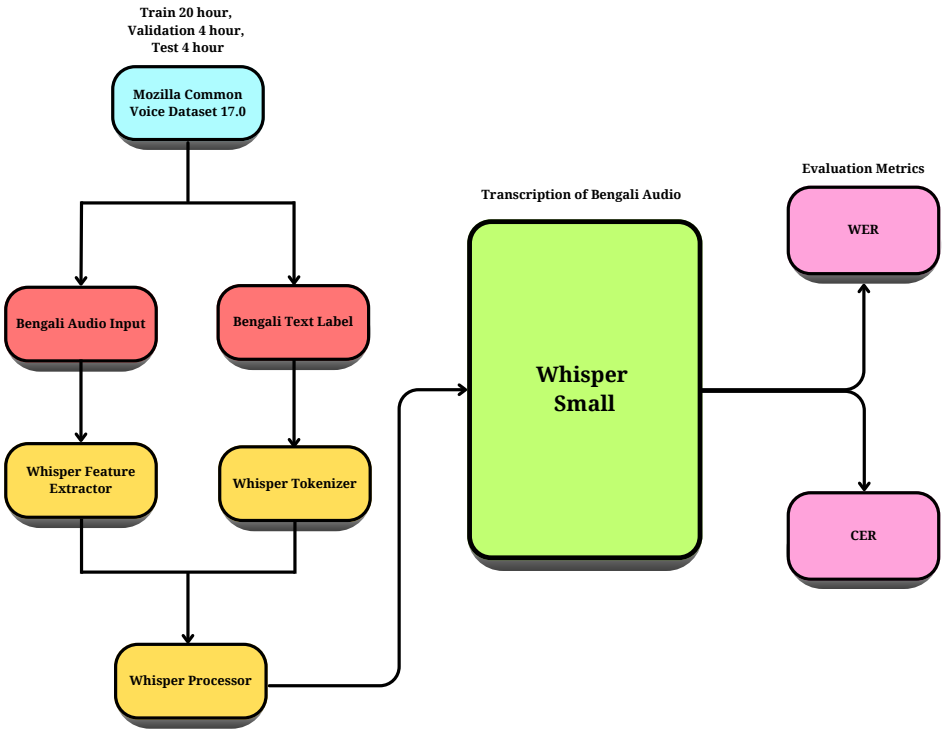

Figure 4.1: Methodology for Transcription Model

The texts were tokenized and unique id was assigned to each token. The tokens are also padded for batch processing requiremnt. Finally, the Whisper model outputs a bunch of tokens(with ID) which are then matched with the label tokens in order to calculate a word error rate(WER) and a character error rate(CER).The model is trained for 2500 steps and evaluated at every 625 steps

4.2 Wav2vec2-xls-r-300

The training process for Bengali ASR using XLS-R Wav2Vec2 began with comprehensive data preprocessing of the 20-hour Mozilla Common Voice Bengali training dataset and 4-hour validation set. Audio files were automatically resampled from their original 48kHz sampling rate to 16kHz as required by the XLS-R model, while text transcriptions underwent Bengali-specific normalization including Unicode NFC standardization, removal of punctuation marks while preserving linguistic meaning, and creation of a character-level vocabulary consisting of approximately 100 unique Bengali characters including conjuncts, diacritics, and special tokens for padding, unknown characters, and word boundaries. The training pipeline utilized the facebook/wav2vec2-xls-r-300m pretrained checkpoint with the CNN feature extraction layers frozen to preserve learned acoustic representations, while a new linear classification head matching the Bengali vocabulary size was added and trained alongside the transformer encoder layers. Training was configured with a per-device batch size of 16 and gradient accumulation over 2 steps for an effective batch size of 32, employing AdamW optimizer with a learning rate of 2e-4, 1000 warmup steps, and regularization through attention dropout (0.1), hidden dropout (0.1), SpecAugment masking (0.075), and layer dropout (0.05).

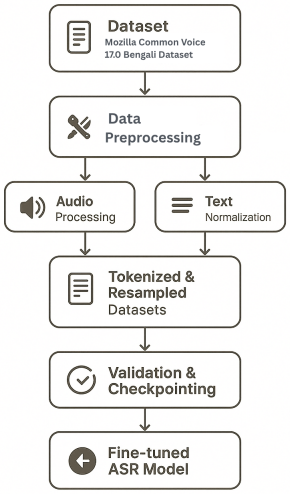

Figure 4.2: Wav2vec2 training workflow

The model was trained using Connectionist Temporal Classification (CTC) loss with mean reduction, dynamic padding to handle variable-length audio sequences efficiently, gradient checkpointing for memory optimization, and FP16 mixed precision training, with evaluation performed every 500 steps using Word Error Rate and Character Error Rate metrics calculated through proper CTC decoding without token grouping, spanning 2500 training steps over approximately 8-9 hours in P100 GPU.

4.3 Real Time inferencing

4.3.1 Dynamic Stride Creation

The Dynamic Stride Creation module uses Silero Voice Activity Detection (VAD) to intelligently determine optimal audio segment boundaries for processing, creating variable-length strides between 1.0 and 2.5 seconds based on natural speech patterns rather than fixed time intervals. The system employs a multi-factor scoring algorithm that analyzes speech activity drops, post-boundary silence quality, and speech ratio changes to identify natural pause points where audio can be segmented without cutting through words. Before applying VAD analysis, the audio undergoes enhancement preprocessing including noise gating, high-pass filtering (150Hz cutoff), and speech band emphasis (300-3000Hz) to improve boundary detection accuracy.

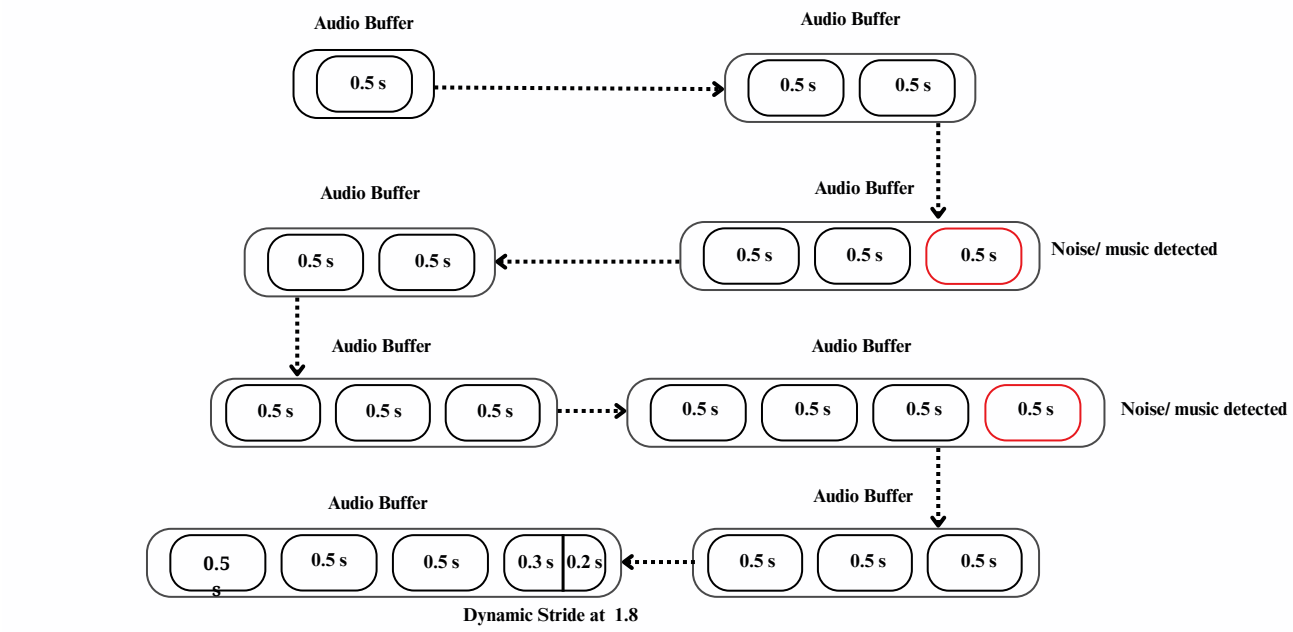

Figure 4.3: Dynamic stride creation process

The scoring system penalizes potential mid-word cuts by detecting moderate speech activity (0.3-0.6 probability) that suggests ongoing speech, while rewarding clear speech-to-silence transitions with scores above 0.20 threshold. This approach ensures that transcription segments align with natural speech boundaries, improving overall accuracy while maintaining efficient processing speeds with fallback mechanisms for challenging audio conditions.

4.3.2 Noise Suppression

The Background Noise Filtering and Suppression module implements a sophisticated multi-stage enhancement pipeline designed to improve speech quality while preserving Bengali phonetic characteristics. The system begins by estimating a noise profile from the quieter segments of the audio (bottom third by energy), then applies gentle spectral subtraction with a conservative alpha value of 1.5 and a high spectral floor (30% of original magnitude) to prevent speech distortion. Speech frequency emphasis targets the 200-3800Hz band that contains fundamental Bengali speech frequencies, using a 70% original and 30% enhanced mix ratio to maintain naturalness. The final stage implements adaptive gain control with 50ms sliding windows to normalize audio levels while limiting maximum gain to 3x to prevent noise amplification.

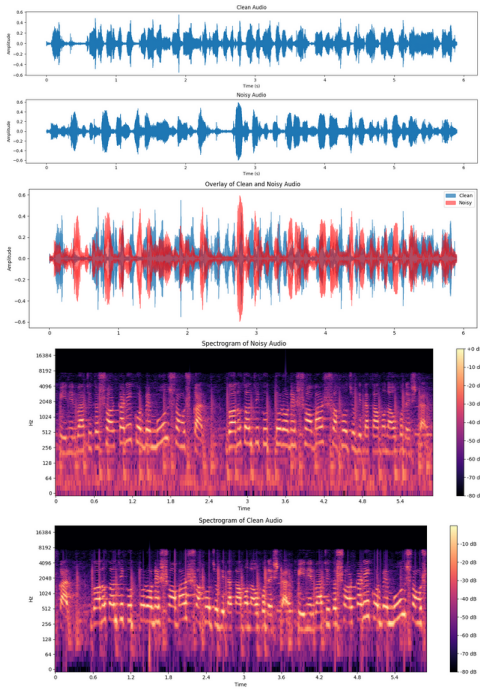

Figure 4.4: Noise Supression

This enhancement pipeline operates on both the original audio for transcription and separately on VAD-optimized audio for better boundary detection, ensuring that noise suppression improves speech recognition accuracy without introducing artifacts that could interfere with the sophisticated overlap detection and stride boundary identification systems.

4.3.3 Two Context inferencing

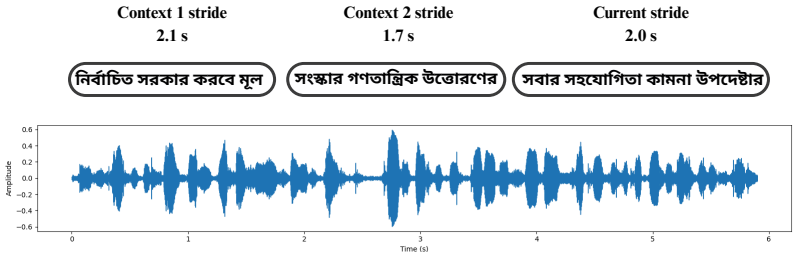

Figure 4.5: Two Context Inferncing

The system employs two distinct context mechanisms** to enhance transcription accuracy. Audio Context combines the current stride with up to two previous enhanced audio strides before Wav2Vec2 inference, creating a 6-7 second audio window that provides the model with cross-stride phonetic information and helps resolve word boundaries that span multiple segments. This audio context is particularly valuable for Bengali's complex consonant clusters and helps the model better understand speech patterns across natural pauses. Text Context operates through the overlap removal system, where the algorithm maintains the previous stride's final 3 words and the complete previous output text to intelligently deduplicate repeated words across processing boundaries. This dual-context approach ensures that while the model benefits from extended audio information for better recognition accuracy, the final output eliminates redundant text through sophisticated sliding window and cross-stride comparison algorithms, resulting in clean, non-repetitive Bengali transcriptions that maintain both linguistic accuracy and processing efficiency.

4.3.4 Overlapping Handling

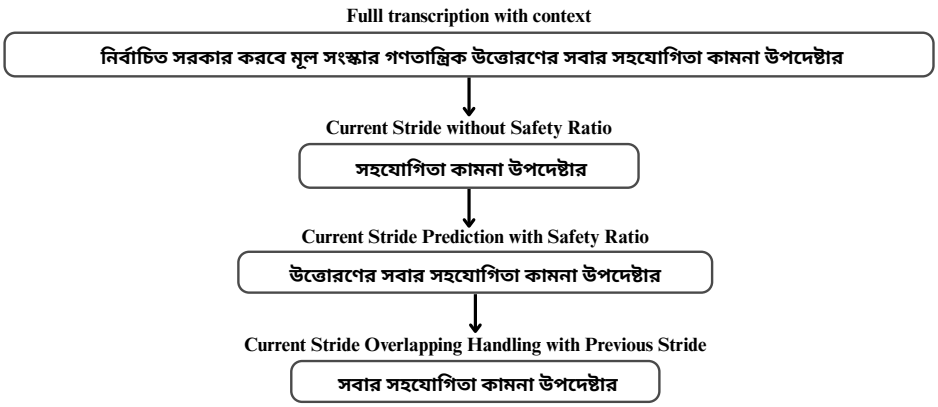

Figure 4.6: Overlapping Handling

The Two-Layer Overlap Handling System solves the critical problem of text duplication that occurs when processing continuous audio in overlapping segments. The first layer uses sliding window deduplication, maintaining the last 3 words from each processed stride and comparing them with the beginning of the next stride to remove exact matches. The second layer performs output overlap deduplication by storing the complete final output from the previous stride and checking if the current output starts with any subsequence from anywhere in the previous output. This dual approach is particularly effective for Bengali speech recognition, incorporating fuzzy matching to handle common spelling variations and ensuring that natural speech boundaries don't create artificial repetitions in the final transcription. The system processes overlaps with 80% character similarity tolerance and maintains minimal memory overhead by storing only essential context information.

Chapter 5

Dataset

5.1 Mozilla Common Voice Dataset

Mozilla Foundation's common voice 17.0 is used as the dataset for our thesis and is acquired from the HuggingFace platform. Common Voice 17.0 contains speech data of 124 languages, including Bengali, the focus of our work. In Samiul et al.(2022), they write that the sentences are either collected from wikipedia or they are added to the dataset manually. Then single channel 48 KHz audio clips are generated by saying the sentences out loud. The dataset consists of 12 columns and approximately 223,000 rows.The audio column consists of Bengali audio samples while the sentence column contains the corresponding sentence representation of the audio. Samiul et al.(2022) also mentions in their paper that a total of 231,120 audio samples are present in the dataset, amounting to 399 hours of total audio data.

5.1.1 Comparison with available dataset

OpenSLR is the another widely used speech to text dataset that is publicly available to us. However, according to Samiul et al.(2022), OpenSLR has 229 hours of audio data which is lower than the 399 hours in common voice. Their paper also mentions that the common voice dataset has approximately 20000 speakers whereas only the voices of 505 people have contributed to the OpenSLR dataset. This indicates that there is more diversity in the common voice dataset than OpenSLR. The paper further clarifies that the recordings in OpenSLR are captured in controlled environments where there is little to no external noise, while common voice does no such thing and has recordings with noise as well. The data in common voice reflects real life situations much better due to it being produced in an uncontrolled environment.

5.1.2 Data Visulization

These four different graphs provide analysis of the Mozila Common Voice Bengali speech corpus characteristics. the audio duration distribution depicts a normal curve with the peak at 5 and 6 seconds. Therefore, most of the samples are between four and eight seconds. It also reveals that the transcripts on average contain 5 to 15 words covering a peak 6 to 8 words. The scatter plot also shed light on the correllation between the audio duration and the text length where there is a positive

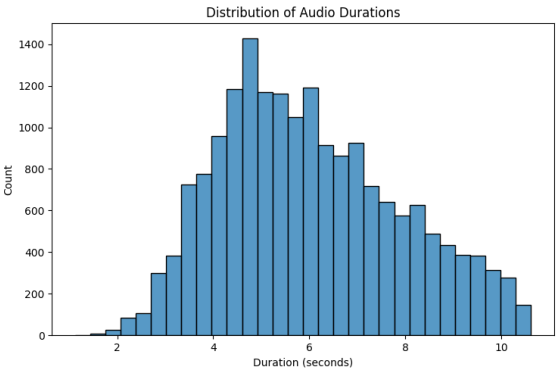

Figure 5.1: Audio Duration Distribution

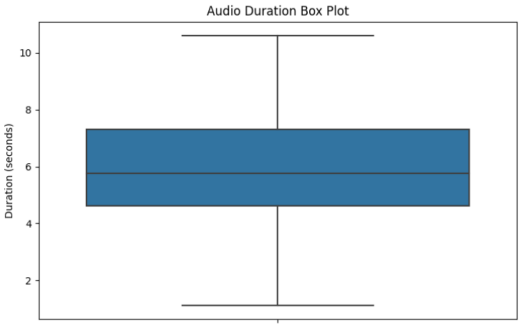

Figure 5.2: Audio Duration Boxplot

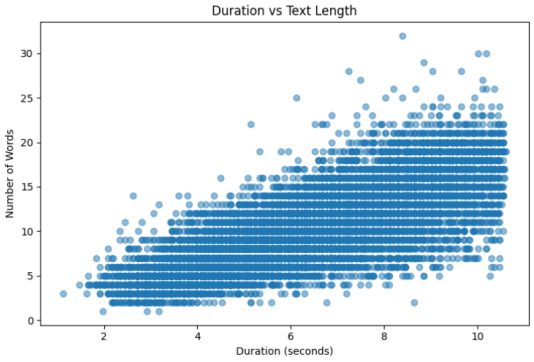

Figure 5.3: Audio Duration vs Text Length Scatter Plot

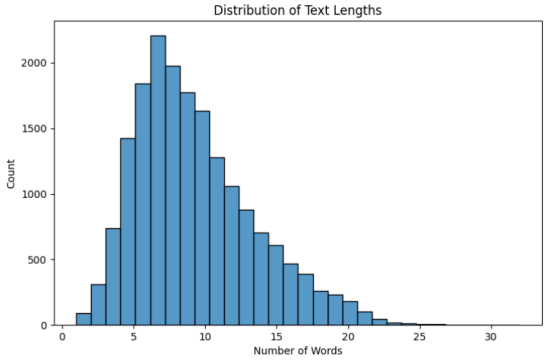

Figure 5.4: Text Length Distribution

relationship showing that longer audio clips contains more words with natural variation. The box plot provides further evidence of the median audio duration being at six seconds with the interquartile range at 4 to 7 seconds. Outliers can be seen to be extenting to 10 seconds. These patterns indicate a more enhanced and well structured dataset which helps train the ASR model since they have consistant audio and text relationships.

5.1.3 Dataset Preprocessing

The dataset is loaded from the HuggingFace datasets library in a streaming format due to memory constraints. Out of the aforementioned twelve columns in the common voice dataset, we have kept the audio and the sentence column only and the other columns are removed. Due to resource constraints, we took 20 hours of audio for training, 4 hours for validation and 4 hours for testing.
The common voice dataset contains 48KHz samples but our deep learning model is trained on 16 KHz samples. So to avoid this incongruence, each audio sample is downsampled to 16KHz. The raw audio samples are then converted to mel spectrograms because of ease of processing the data for the model. Key information is retained from the audio while redundant ones are discarded through this process.Finally, each sentence in the label is broken down into tokens and each token is given an ID. We have also made sure each sentence is of the same length and each audio sample is of the same length by padding.

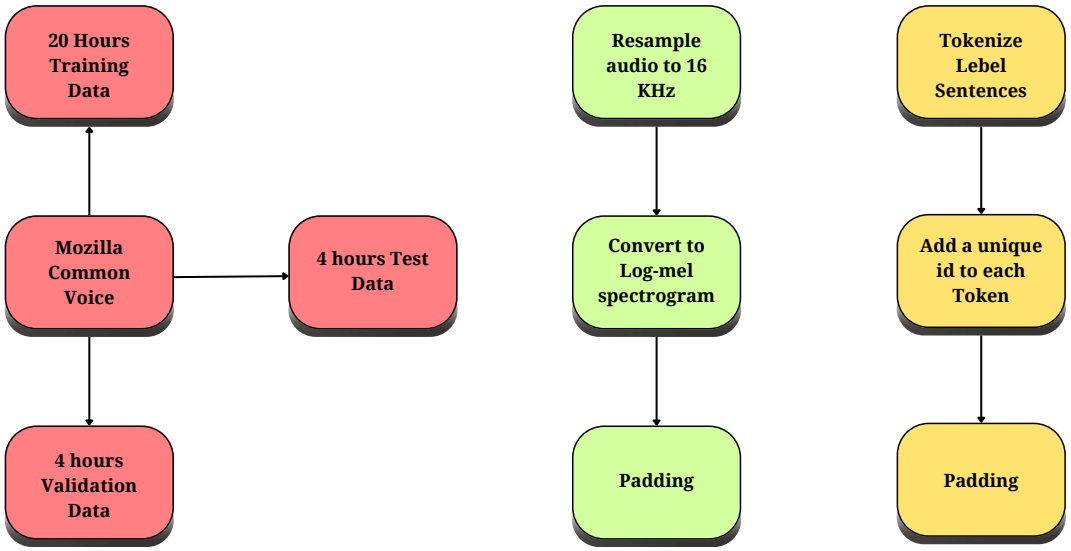

Figure 5.5: Dataset Split and Preprocessing

5.2 Bengali Multimedia Dataset

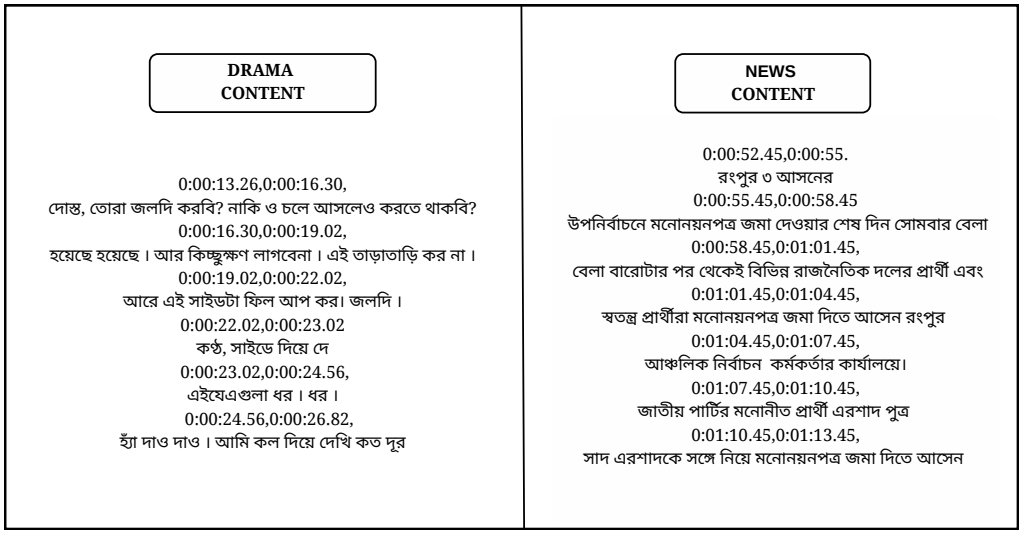

Figure 5.6: Bengali Multimedia Dataset

To evaluate our system for Real Time Closed Captioning for Bengali Multimedia, we created our custom dataset for four hours of Bengali audio and text data. We are using this dataset as our test data, which will show the performance in the real world uses of Bengali contents, mostly focusing on the media which are vastly used among the Bengali spectators. That is why we took two hours of data from news broadcasts and two hours of data from drama contents. Another reason for choosing these two contents is that they include both standard and dialect Bengali speech. We used Voice Activity Detection (VAD) algorithm for the segmentation of audio to ensure speech boundaries and also split data in manageable clips. Because VAD can occasionally cut words in half, we manually reviewed and fixed these instances when they occurred. We manually transcribed each data in each segment in the particular

time intervals. We ensured high accuracy in this process of creating this dataset. Importantly, this dataset fully consists of Bengali audio where both male and female speakers are present. Besides, in the news broadcasts, children also gave interviews which are actually covering the voices of all ages. There are various background noises for example music, protest and many other extra noises are present. This will also evaluate our model's performance on real-life content, particularly how it handles issues such as overlapping speech, background music and noises. Ultimately, this approach makes our evaluations both realistic and meaningful.

Chapter 6

Model

6.1 Whisper

This research used Whisper model for training audio/video content. Whisper models are used for high accuracy multilingual tasks. This model is pre-trained with labelled data and can be fine tuned. Its end to end architecture is designed in a way that it can process data directly and can handle real-world noise variations. Whispers streaming ASR compatibility is a feature that allows it to work with real time transcription rather than waiting for the entire audio file to be processed. Besides, audio chunking, support for multi-language and accents, open source and flexibility etc features are provided by whisper.

6.1.1 Description of Whisper

The whisper model, developed by openAI, is a pre-trained model for automatic speech recognition (ASR) and speech translation. It is a powerful and versatile tool for ASR related tasks. This model is trained on 6,80,000 hours of labelled data. Generalising to many datasets without fine tuning is the powerful ability of this model. Importantly, this model is also an encoder-decoder model based on a transformer. Besides, this model is also called sequence to sequence model. While training this model, large-scale weak supervision is used for speech data annotation. Whisper models are pre-trained in five different sizes. Each size has different complexity and capability.

Size	Parameters	Support	Purpose
Tiny	39 million	Trains both English-only and multilingual	Suitable for lightweight tasks where lower computational resources are required.
Base	74 million	Supports both English-only and multilingual	Suits lightweight tasks but provides better accuracy than the tiny size.
Small	244 million	Supports both English-only and multilingual	Offers good accuracy and efficiency.
Medium	769 million	Supports both English-only and multilingual transcription	Ensures higher accuracy for more challenging tasks.
Large	1.55 billion	Multilingual only	Ideal for the highest accuracy in multilingual transcription and translation tasks.

Table 6.1: Overview of Whisper Model Sizes and Features

Importantly, along with multilingual support, this model can handle background noise, challenging audio conditions which is considered as one of the important features. The versatility of the whisper model denotes its amazing performance for

ASR, language detection etc. Besides, this model can achieve better transcription quality because of its high accuracy even in the noisy or low-resource audio. Moreover, there are many applications of Whisper. For example, speech-to-text Services, Language learning , Real-time transcription Integration with other systems etc are mentionable. But this model has some limitations as well. For example, Depending on the model size and hardware, the model processes slowly. Besides, the model may struggle with low-resource languages or extremely noisy environments.

6.1.2 Architecture of Whisper

Whisper model is considered as an advanced Automatic Speech Recognition (ASR) model which is mainly designed for ensuring high accuracy while converting speech to text , translation and multilingual transcription. This model is built on a sequence-to-sequence Transformer architecture.

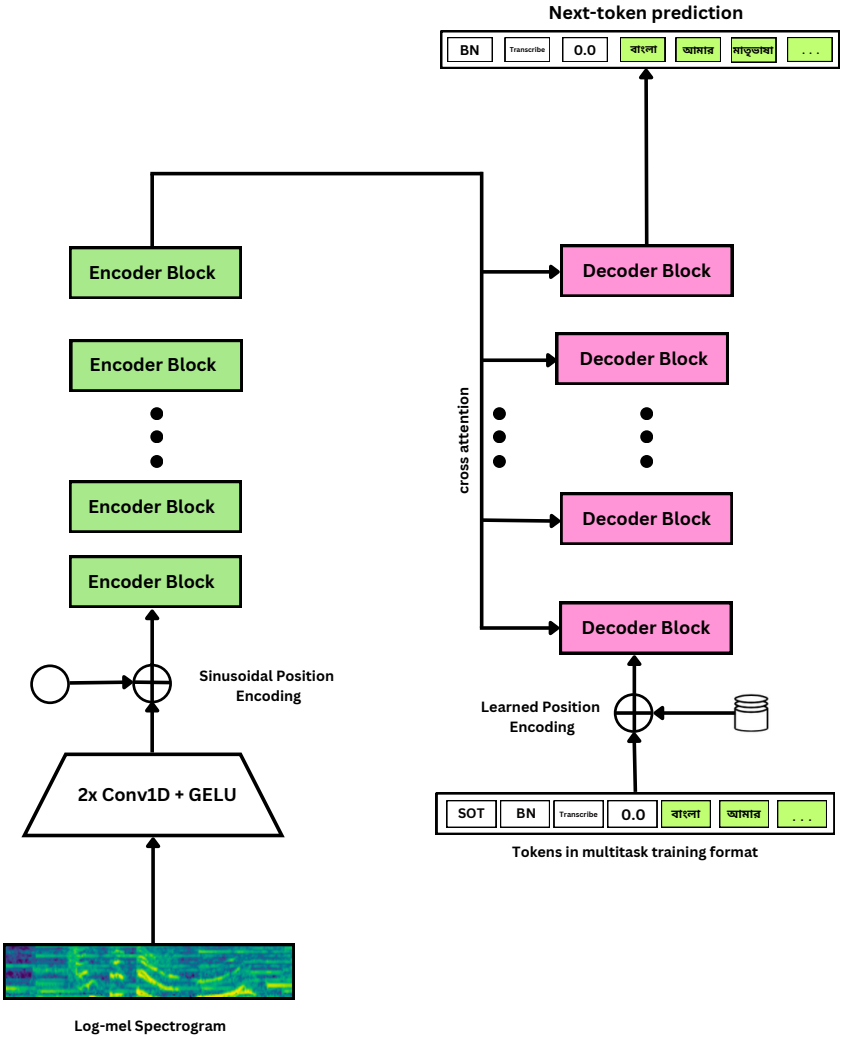

Figure 6.1: Whisper Architecture

In the whisper model, there are two main components. For processing audio inputs, an encoder is present. For generating transcriptions, a decoder is used. For creating visual representations of audio frequencies over time, the input is first converted into a log-Mel spectrum. By using 1 dimensional convolutional layers (Conv1D) along with GELU activation, preprocessing of this spectrogram is occured. This step helps to identify local patterns in the speech signal. Consequently, for saving the structure of input, before passing the processed data through multiple Transformer encoder blocks, sinusoidal positional encoding is added. This block of encoders are used to capture a significant range of dependencies which enables the model to understand complex audio sequences. After the transformation of speech into a rich feature representation, decoder takes the charge and receives tokenized text input, including Start of Transcript (SOT), language tags and previous text tokens that helps in providing contexts. Decoder passes those tokens through multiple Transformer decoder blocks but before doing that, it applies learned positional encoding to these tokens. These decoder blocks align the text tokens with encoded speech features so that the model can predict the next token in the transcription sequence. This architecture of whisper allows it not only to transcribe speech but also translate it across languages which shows its ability to perform multitask learning and speech recognition with utmost efficiency. By transcription pipelining, the second phase of whisper's workflow starts. In this stage, to establish context and improve accuracy, the model processes previous text tokens. Language identification is done by the model before starting transcription. For No Speech, the output of the system automatically shows "No Speech." On the contrary, when speech is detected, the model performs direct transcription (X→ X) depending on the language setting. Besides, the whisper model can transform time aligned transcription which is considered as one of the unique features of this model. In this process, the model assigns precise timestamps to each generated text tokens. Because of this, accurate synchronization of transcribed audio is enabled which makes it ideal for applications like subtitles and automated video captioning. When the model encounters an End of Transcript (EOT) token, the transcription process stops.

6.2 Wav2vec2-xls-r-300m

This research also used Facebook AI's wav2vec2-XLS-R-300M which is part of the Cross-Lingual Speech Representation family. As a result, it poses the capability to engage in robust speech translation, automated speech transcription and any similar activities related to speech but pertaining to a large number of languages. Contrary to Whisper, it is pretrained using a semi supervised learning approach so no manual transcriptions are required.

6.2.1 Description of Wav2vec2-xls-r-300m

The Wav2Vec2-XLS-R- 300m, developed by FacebookAI, is based on the wav2vec2 2.0 model. It is a multilingual extension of wav2vec2 2.0 trained on 128 languages with the help of datasets such as BABEL, Commonvoice and Multilingual librispeech. A total of 436,000 hours of unlabeled speech data was used to pretrain the model. The model can be used for large scale operations like automated speech

recognition, speech translation or identification of a particular speaker across multiple languages due to the robust and scalable speech recognition capabilities of the model.

6.2.2 Architecture of Wav2vec2-xls-r-300m

The model consists of a convolutional neural network feature encoder to convert the raw waveform into hidden speech representations that contain the acoustic features of the sound. The output of the encoder is then passed onto a multilayer transformer with masking. The masking is used during pre-training where certain parts of the input are masked so that the transformer can learn to predict those masked bits using the surrounding context. Looking further into the pretraining process, the latent

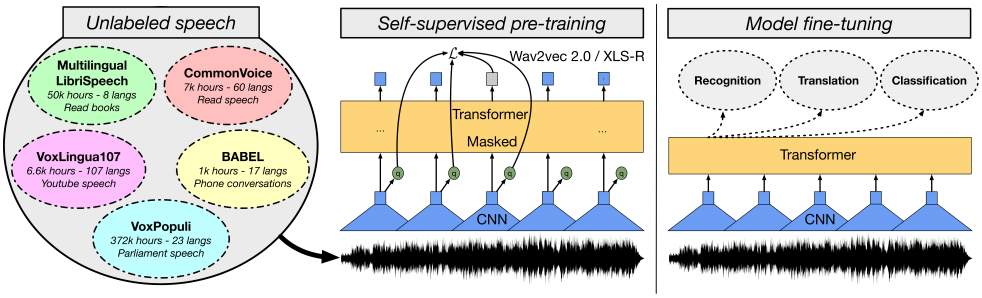

Figure 6.2: Wav2vec2-xls-r-300m Architecture

speech representations created by the CNN encoder are broken down into small discrete pieces using a quantization module. This allows the model to predict the masked parts of the audio. Once the pretraining process is complete, the model can be fine tuned for speech recognition, machine translation or speaker identification. The pretraining process is end to end and self supervised learning, meaning the model does not required manual transcriptions or labeled data for training. A contrastive loss function is used during the pretrained process so that the correct inputs in the masks can be obtained among all the other distractions.

6.3 Wav2vec2-xls-r-300m-bengali-commonvoice

Facebook's `wav2vec2-xls-r-300m` model was fine-tuned on the Bengali dataset from `OPENSLR SLR53` to create the `arijitx/wav2vec2-xls-r-300m-bengali` model, which is available on Hugging Face. On the evaluation set, a Word Error Rate (WER) of 0.217 and a Character Error Rate (CER) of 0.0473 were observed without using a language model. When a 5-gram language model, trained on 30 million random sentences from the `AI4Bharat IndicCorp` dataset, was introduced, the WER dropped to 0.153 and the CER to 0.0341. On the test set from `OPENSLR`, a WER of 0.217 and CER of 0.047 were also observed without a language model. With the language model applied, the test scores improved further, reducing the WER to 0.153 and the CER to 0.034. The wav2vec2-xls-r-300m-bengali was further fine-tuned on the common voice 9.0 bengali dataset to create the shahruk10/wav2vec2-xls-r-300m-bengali-commonvoice, also available on huggingface. A total of 300 hours of Bengali Audio data was used for the fine tuning process. With a 5-gram language model, test

WER and CER were 0.017939 and 0.08079 respectively, a significant improvement from the previously discussed model.

6.4 BanglaASR

The used the Mozilla Common Voice Bengali dataset v9.0 , which contains approximately 400 hours of Bengali speech. This dataset is very useful for real-life situations as this holds the voices of both male and female of native speakers from different regions. BanglaASR which is avaiable in huggingface has used this dataset. BanglaASR is afine-tuned Whisper-small model which is developed by OpenAI . It is fine-tuned by using 40,000 training samples and 7000 validation samples. It is taken from a 400 hour Mozilla Common Voice dataset. For 12000 steps , the model was trained and 4.58% WER (Word Error Rate) has been achieved. The BanglaASR shares preprocessing scripts to clean and segment the audio data, a training pipeline for fine tune and evaluation code for calculating CER AND WER which are very useful tools and scripts.

Chapter 7

Evaluation

7.1 Bengali Audio to Text Transcription Metrics

.

7.1.1 Character Error Rate

Character Error Rate (CER) is a metric used to evaluate speech recognition accuracy by comparing predicted text with reference text at the character level. It quantifies transcription errors by counting the number of incorrect characters relative to the total character count.

$$\text{CER} = \frac{S + D + I}{N} \times 100\% \tag{7.1}$$

where:

- S: Number of character substitutions
- D: Number of character deletions
- I: Number of character insertions
- N: Total number of characters in the reference text

7.1.2 Word Error Rate

Word Error Rate (WER) is the standard metric for evaluating automatic speech recognition systems, comparing predicted transcriptions with reference text at the word level. It provides a comprehensive measure of transcription accuracy across different types of recognition errors.

$$\text{WER} = \frac{S_w + D_w + I_w}{N_w} \times 100\% \tag{7.2}$$

where:

- S_w: Number of word substitutions
- D_w: Number of word deletions

- I_w: Number of word insertions
- N_w: Total number of words in the reference text

For Bengali ASR evaluation, both metrics are essential as CER captures fine-grained character-level errors particularly important for conjunct consonants, while WER provides semantic-level accuracy assessment. Lower values indicate better performance, with 0% representing perfect transcription accuracy.

7.2 Results

7.2.1 Whisper-Small

In our research, the Whisper model is used to train the mozilla common voice dataset for Bengali audio data. For training, 2500 steps are performed along with evaluation for every 625 steps, which shows the constant better result for both Word Error Rate (WER) and Character Error Rate (CER).
In the case of WER , along with the increment of epoch number , results are getting better. After the first 625 steps, we got Training loss 0.2234 and validation loss 0.2063. Besides , we got Word Error Rate 54.841 and Character Error Rate 17.233. When the next 625 steps run, for 1250 steps, we got the data better than the previous case. We went from WER 54.84% to 42.99%. Similarly, for CER , the error rate is decreasing from 17.23% to 12.92%. This improvement continues in the next 625 steps as well. Validation improved from initial 0.206350 to 0.133266 after 1875 steps. This scenario is applicable for WER as well as the error rate decreased from 42.99% to 39.29% and Character error rate decreased from 12.92% to 11.877%. As we run for 2500 steps, we are getting the final validation loss 0.0678 along with training loss 0.126156. Importantly, we are getting better results for WER and CER which are 37.44% and 11.32% respectively. Because of the steady Training loss from 0.223 to 0.068 when validation loss is improving from 0.206 to 0.126. the model is converging without overfitting. For test data, we finally evaluated 4 hours of unseen data and we are getting WER 39.33% and CER 12.45% . We had to overcome

Step	Training Loss	Validation Loss	Wer	Cer
625	0.2234	0.2063	54.841	17.233
1250	0.1281	0.1464	42.995	12.921
1875	0.0754	0.1332	39.295	11.877
2500	0.067800	0.1261	37.440	11.323

Table 7.1: Training and Validation Metrics

certain obstacles to reach our outcome. One such obstacle was the lack of resources at our disposal which did not allow us to run more epochs because of the hours of data we had loaded into our already limited memory. Further, due to a limit in the session timer, it was difficult for us to train the model on more epochs . Moreover, it took substantially long to train the model with the given resources and so it proved quite challenging for us to play around with the number of epochs or some of the hyperparameters. However, we can train for bigger data and run for more epochs, our word error rate and character error rate will be improved and we will get better results.

7.2.2 Wav2vec2-xls-r-300m

Along with whisper-small we have also have trained wav2vec2-xls-r-300m on the 20 hour data collected from mozila common voice bengali. We have used the similar setting as whisper small to calulate our training loss, validation loss, wer and cer to show the fine tuning progress. In the case of WER , along with the increment of epoch number , results are getting better. After the first 625 steps, we got Training loss 0.5543 and validation loss 0.5729. Besides , we got Word Error Rate 68.42%

Step	Training Loss	Validation Loss	Wer	Cer
625	0.5543	0.5729	68.425	26.762
1250	0.2829	0.2643	55.891	19.284
1875	0.1692	0.2161	44.156	18.738
2500	0.1437	0.1874	41.359	17.973

Table 7.2: Training and Validation Metrics

and Character Error Rate 26.76%. When the next 625 steps run, for 1250 steps, we got the data better than the previous case. We went from WER 68.42% to 54.89%. Similarly, for CER , the error rate is decreasing from 26.76% to 19.28%. This improvement continues in the next 625 steps as well. Validation improved from initial 0.5729 to 0.2161 after 1875 steps. This scenario is applicable for WER as well as the error rate decreased from 54.89% to 44.15% and Character error rate decreased from 19.28% to 18.74%. As we run for 2500 steps, we are getting the final validation loss 0.1974 along with training loss 0.1437. Importantly, we are getting better results for WER and CER which are 41.35% and 17.97% respectively.For test data, we finally evaluated 4 hours of unseen data and we are getting WER 44.63% and CER 20.97% .

7.2.3 Real time closed captioning result

Model Name	**WER**	**CER**	**Inference time**
BanglaASR	49.83%	27.57%	2.47
Wav2vec2-xls-r-300m-bengali-commonvoice	34.67%	15.92%	0.44

Table 7.3: Mozilla Common Voice 17.0 ASR Model Performance Comparison

We have tested our data on three types of test data. We tested irst using 4 hours of Mozilla Common Voice 17.0. As most of the test data is standardized Bengali the model showed better results than our own dataset. Wav2vec2-xls-r-300m-bengali-commonvoice showed a CER of 15.92% and WER of 34.67%. It would have shown better result if overlapping text handling were better. On the hand, BanglaASR showed worse result than Wav2vec2-xls-r-300m-bengali-commonvoice as it hallucinated when the audio contained small words at the end of current stride. After testing on mozila we also tested on two seperate content of our dataset separately. While performance on both were worse than mozila dataset, the results went really down on the drama content of the dataset. The reasons were fast paced speech, too much noise, different accents, too many overlapping speech, frequent change in emotion along with short length speech is affecting the models heavily.

Model Name	WER	CER	Inference time
BanglaASR	72.48%	38.56%	2.64
Wav2vec2-xls-r-300m-bengali-commonvoice	63.29%	32.81%	0.53

Table 7.4: News Content ASR Model Performance Comparison

Model Name	WER	CER	Inference time
BanglaASR	93.47%	70.35%	3.43
Wav2vec2-xls-r-300m-bengali-commonvoice	83.95%	66.73%	0.94

Table 7.5: Drama Content ASR Model Performance Comparison

Due to hallucination of BanglaASR it becomes much worse. The news content shows better result due to consistent tone and accent of the reporters. But when general people are speaking it becomes worse due local dialect speech, multiple accent and background noise. There is also incosistency in the drama content wer and cer ratio which is less than 1.5. Due to paragraph to paragraph evaluation, this inconsistency can be seen. Hence, a better evaluation technique should be used to get proper result.

Chapter 8

Limitations and future works

8.1 Limitations

While our system provides sub second latency we faced some real issues when we tried to test with real world multimedia contents.

8.1.1 Voice Overlapping and Background Noise

In drama content, we found there were lot of overlapping audios. The system couldn't perform well when it faced such scenarios. It made gibberish trancriptions. It also affected the next transcriptions when used as context for both audio and texts. Moreover, our noise suppresion system suppresses noise in less noisy areas but in extreme noisy audios the transcription is almost wrong.

8.1.2 Mid Word Cutting

Using silero vad, we tried to make dynamic stride by finding word boundary in small gaps. It works well in many cases but in some cases in cuts word in the middle. This mid word cutting makes overlapping detection harder and resulting in many overlaps with incomplete word and complete words.

8.1.3 Local Dialect Speech and Mordern Accent

The models we used are trained on mozila common voice. Most of the audios in this dataset are for standard bengali. Hence, the model struggles when faced with local dialect. Moreover, most of the drama contents nowadays follows a mordern accent which the mozilla common voice dataset completely lacks. As a result, the results go downhill while testing on drama contents.

8.1.4 Emotional Speech Degradation

The system shows significant performance degradation with emotional speech patterns, which are common in multimedia content specially entertainment shows like drama contents. Strong emotions including anger, excitement, sadness, and fear substantially alter fundamental speech characteristics including pitch variation, speaking rate, voice quality, and articulation clarity. Emotional states cause irregular

rhythm and timing patterns that particularly affect Bengali tonal variations and stress placement, which are crucial for accurate conjunct consonant identification and word boundary detection.

8.2 Future Works

As we studied the limitations, we have found some future works that might make the transcription quality better. This will result in more accurate subtitles and take our system closer to real world implementation.

- Context aware overlapping handling
- Training model for noisy data
- Training model on real life multimedia contents
- Training VAD for Bengali to make better strides
- Create separate processing pipelines for different speakers in multi-speaker scenarios
- Implement dynamic parameter adjustment based on content type detection

Chapter 9

Conclusion

To sum up, building a closed captioning system for Bengali content is a type of work which is not explored that much yet. So, definitely, it is not only challenging work technically but also very crucial for making digital content more accessible and inclusive. In our research, we also get to know the clear gap in Bengali ASR technology. We, in this research, aim to meet these gaps by leveraging deep learning techniques such as Transformer-based models or hybrid models. We added a Gantt Chart to demonstrate our work plan for the entire thesis. This work has a huge potential to open new avenues for linguistic inclusivity in technology. We believe that our work can help the people who are deaf or have hearing issues. We also believe this work can be a landmark for those researchers who want to work in this topic in future and our work will provide tangible benefits for Bengali speaking individuals.

Bibliography

[1] I. González-Carrasco, L. Puente, B. Ruiz-Mezcua, and J. L. López-Cuadrado, "Sub-sync: Automatic synchronization of subtitles in the broadcasting of true live programs in spanish," *IEEE Access*, vol. 7, pp. 60 968–60 983, 2019. DOI: 10.1109/ACCESS.2019.2915581.

[2] A. Kannan, A. Datta, T. N. Sainath, *et al.*, *Large-scale multilingual speech recognition with a streaming end-to-end model*, 2019. arXiv: 1909.05330 [eess.AS]. [Online]. Available: https://arxiv.org/abs/1909.05330.

[3] N. Moritz, T. Hori, and J. L. Roux, "Triggered attention for end-to-end speech recognition," in *ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2019, pp. 5666–5670. DOI: 10.1109/ICASSP.2019.8683510.

[4] E. Demirel, S. Ahlbäck, and S. Dixon, "Automatic lyrics transcription using dilated convolutional neural networks with self-attention," in *2020 International Joint Conference on Neural Networks (IJCNN)*, 2020, pp. 1–8. DOI: 10.1109/IJCNN48605.2020.9207052.

[5] N. Moritz, T. Hori, and J. Le, "Streaming automatic speech recognition with the transformer model," in *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2020, pp. 6074–6078. DOI: 10.1109/ICASSP40776.2020.9054476.

[6] X. Sun, Q. Yang, S. Liu, and X. Yuan, "Improving low-resource speech recognition based on improved nn-hmm structures," *IEEE Access*, vol. 8, pp. 73 005–73 014, 2020. DOI: 10.1109/ACCESS.2020.2988365.

[7] M. Á. Tündik, B. Tarján, and G. Szaszák, "A low latency sequential model and its user-focused evaluation for automatic punctuation of asr closed captions," *Computer Speech Language*, vol. 63, p. 101 076, 2020, ISSN: 0885-2308. DOI: https://doi.org/10.1016/j.csl.2020.101076. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0885230820300097.

[8] A. Martín García, I. Gonzalez-Carrasco, V. Rodriguez-Fernandez, M. Souto, D. Camacho, and B. Ruíz-Mezcua, "Deep-sync: A novel deep learning-based tool for semantic-aware subtitling synchronisation," *Neural Computing and Applications*, pp. 1–15, Feb. 2021. DOI: 10.1007/s00521-021-05751-y.

[9] Y. Shi, Y. Wang, C. Wu, *et al.*, "Emformer: Efficient memory transformer based acoustic model for low latency streaming speech recognition," in *ICASSP 2021 - 2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2021, pp. 6783–6787. DOI: 10.1109/ICASSP39728.2021.9414560.

[10] J. Jorge, A. Giménez, J. A. Silvestre-Cerdà, J. Civera, A. Sanchis, and A. Juan, "Live streaming speech recognition using deep bidirectional lstm acoustic models and interpolated language models," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 30, pp. 148–161, 2022. DOI: 10.1109/TASLP.2021.3133216.

[11] M. Priya, K. R. Dhanaraj, L. A. Kumar, and S. Rose, "Multilingual low resource indian language speech recognition and spell correction using indic bert," *Sādhanā*, vol. 47, Nov. 2022. DOI: 10.1007/s12046-022-01973-5.

[12] S. Chen, X. He, L. Guo, *et al.*, *Valor: Vision-audio-language omni-perception pretraining model and dataset*, 2023. arXiv: 2304.08345 [cs.LG]. [Online]. Available: https://arxiv.org/abs/2304.08345.

[13] S. Chen, H. Li, Q. Wang, *et al.*, "Vast: A vision-audio-subtitle-text omni-modality foundation model and dataset," in *Advances in Neural Information Processing Systems*, A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine, Eds., vol. 36, Curran Associates, Inc., 2023, pp. 72 842–72 866. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2023/file/e6b2b48b5ed90d07c305932729927781-Paper-Conference.pdf.

[14] Y. Fathullah, C. Wu, E. Lakomkin, *et al.*, *Prompting large language models with speech recognition abilities*, 2023. arXiv: 2307.11795 [eess.AS]. [Online]. Available: https://arxiv.org/abs/2307.11795.

[15] D. Macháček, R. Dabre, and O. Bojar, *Turning whisper into real-time transcription system*, 2023. arXiv: 2307.14743 [cs.CL]. [Online]. Available: https://arxiv.org/abs/2307.14743.

[16] J. Wu, Y. Gaur, Z. Chen, *et al.*, "On decoder-only architecture for speech-to-text and large language model integration," in *2023 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, 2023, pp. 1–8. DOI: 10.1109/ASRU57964.2023.10389705.

[17] A. M. Samin, M. H. Kobir, M. M. S. Rafee, *et al.*, "Banspeech: A multi-domain bangla speech recognition benchmark toward robust performance in challenging conditions," *IEEE Access*, vol. 12, pp. 34 527–34 538, 2024. DOI: 10.1109/ACCESS.2024.3371478.