Whisper for Bengali Regional Dialect Automatic Speech Recognition System

A F M Mahfuzul Kabir†, S K Nahin †, Sabbir Hossain Ujjal†
Machine Learning Engineer
Advanced Chemical Industries Limited
Dhaka, Bangladesh
†These authors contributed equally to the work

***Abstract*—Automatic speech recognition (ASR) systems refer to automatic transcription of speech data to text in the respective language. While the ASR systems for the Bangla language has developed at a good scale in recent years, the regional dialects still remains a problem to be solved. Bangla language is spoken by over 300 million people in the whole world, while many regional dialects exist all over Bangladesh and India. In this work, we introduce a comprehensive analysis of OpenAi's Whisper and Facebook's Wav2Vec2 based XLS-R models on Bangla regional dialect audio dataset. The analysis was done for the Kaggle competition 'ASR for Regional Dialects' hosted by Islamic University of Bangladesh. Our analysis shows an improvement of a good margin over the baseline with 70% WER with Whisper medium, 84% WER with Wav2Vec2 and 75% WER with Whisper combined with LORA for parameter efficient fine-tuning. Further thresholding with duration for each token involving word level timestamps for speech data using Whisper and Wav2Vec2 models together resulted in a final score of 67.5% WER in the private leaderboard and 68.3% WER in the public leaderboard of Kaggle competition.**

***Index Terms*—Automatic Speech Recognition, Deep Learning, Whisper, Wav2Vec2, LoRA**

I. Introduction

The field of Automatic Speech Recognition (ASR) has been improving a lot in recent years, making it one of the most crucial applications of deep learning. With release of newer and better models almost every couple of years, this field is running fast towards the promise of voice-automated services of the future. While the advancement is true for most of the high resource languages like English, French and others, the scenario is a bit different for Bengali ASR systems. Bengali, being the seventh most spoken language in the world, is surprisingly a low-resource language due to the lack of data. Platforms such as CommonVoice [1] and BengaliAi [] have been working towards minimizing the gap between the current state of AI and Bengali language.

In early stages of Bengali ASR systems, which were not long ago, the Facebook's wav2vec2 [2] based models dominated a lot. The introduction of Convolutional Neural Network (CNN) based feature extraction and recognition of speech using transformer based networks and CTC loss [3] proved to be very effective for low resource languages, such as Bengali. In [4], the authors developed an ASR system which surpassed all of the previous bechmarks of that time, with an WER of 14% in CommonVoice Bengali dataset. Later in [5], the authors surpassed even the previous one with an WER of 12.8% on the same benchmark by introducing a newer vocabulary with the same Wav2Vec2 based architecture. At last, in the 2023 Kaggle competition for Bengali speech recognition [6], the team **Chimege** [7] achieved the best out-of-distribution score for Bengali ASR system using Whisper. Since then, OpenAi's Whisper [8] has been the default Bengali ASR solution for most of the systems developed.

However, the Bengali language is itself a complex language with a huge number of native speakers all over the world. Specially, the regional dialects of native Bengali speakers are not only a challenge yet to be addressed, but also a huge obstacle towards a robust and efficient ASR system. In these paper, we analyze the effect of fine-tuning Whisper and Wav2Vec2 [2] models with Bengali regional dialect dataset by BengaliAi. The Whisper model was trained both with parameter efficient fine-tuning method known as LORA and the normal training method. The results show a huge improvement over the baseline models on the dataset, while the Whisper medium without LORA(Low Rank Adaptation) [9] promises to be the best one for the task.

II. Methodology

The methodology can be divided into four major sections, 1) Data exploration & Preprocessing, 2) Augmentation, 3) Training and 4) Inference. All of these sections were explored carefully, we can explain the methodologies below.

A. Data Exploration

The dataset provided by BengaliAi contained audio data from different regional dialects of Bangladesh and India. The regions were Barishal, Chittagong, Habiganj, Kishoreganj, Narail, Narsingdi, Rangpur, Sylhet, Sandwip and Tangail.The dataset was collected on various day-to-day life event topics from 373 individuals and total duration of speech corpus is over 79+ hours. The audio data contained 16kHz speech samples from all of these regions and mostly were between 15 to 20 seconds long. The labels or the transcriptions were text data, unnormalized and contained punctuations. The regional distribution wasn't a balanced one, with samples from Sylhet having the highest number of samples. The data was explored further to figure out the number of valid words, that is words without punctuations to arrive in the whole training dataset.

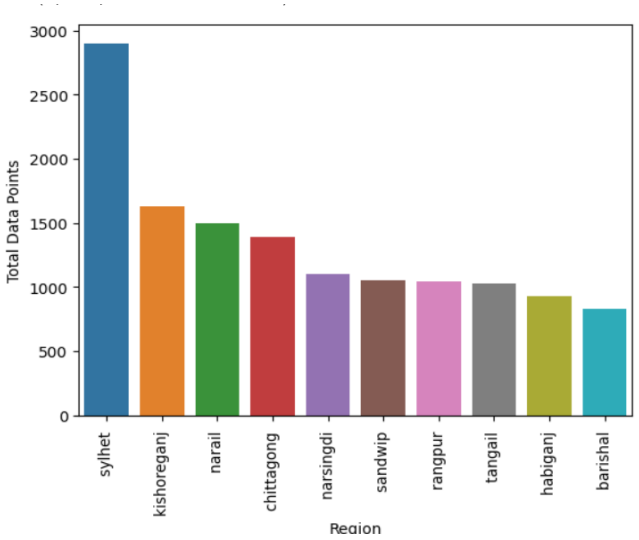

Fig. 1: Region Wise Data Distribution

To our surprise, the most frequent word in the training dataset was the symbol '<>' with more than 10,500 appearances. The second most frequent word had only over 7000 appearances in the dataset. The '<>' symbol represented the portions where the audio wasn't interpretable by human. Since that can also happen in the test dataset or in real life situations, we decided not to remove this symbol for training our model.

B. Preprocessing

The preprocessing of the dataset involved processing the text data for a smooth training of the ASR models. The audio data was mostly processed, so we did not apply any preprocessing for them. The following preprocess options were applied.

- The text data was normalized using banglanlptoolkit. The normalizer in this library uses bnunicodenormalizer to normalize the unicode values for text data and remove any English words.
- The punctuations in the texts were tested by removing and keeping them in the dataset. The transformer models can obviously predict punctuations as well as a token, but for long it has been a normal practice to remove them since they don't contain any acoustic feature. However, models like Whisper can predict punctuations pretty well, so it was a matter of test to see if the model performs better with punctuations or not.
- The uninterpretable token '<>' was removed for the first few trials. But it was impossible to use any deep learning based models to manually enter the token for the test set, so we kept that for the latter part of the training.
- Split the refined dataset into train, validation dataset ensuring proper distribution of each region on both dataset so that the model learns all of the region properly.

C. Augmentation

The dataset was full of noisy speech data. From our previous experience with ASR, removing noise from audio data usually doesn't result in a good score. Also, in practical aspects, the use of denoiser models can result in unwanted artefacts in the audio signals and also more inference time, which is not practical for real life use cases. So, we didn't use any denoiser, rather we augmented the training data as much as possible for the model to learn the noise in the training data. The augmentation techniques included speed augmentation, reverb augmentation, volume augmentation, noise augmentation with white, pink and other noise signals and etc. Our augmentation technique was applied randomly with the audio files depending on some previously set probability. Besides these, we also added another noise augmentation with a huge dataset of natural sounds. The idea was to include environment noises randomly to the audio samples for the model to learn them as well. Overall, these augmentation techniques are supposed to improve the overall model robustness, efficiency and quality.

D. Training

The training of our system can be divided into three parts, for three major training processes.

1) Wav2Vec2 Training: In our work we first tried combination of Acoustic model (AM) and a Language model (LM). AMs take acoustic features and predict a set of letter units. The LM assigns probabilities to word sequences. And then Connectionist Temporal Classification (CTC) loss is calculated against text features. We have used Wav2Vec2.0 [2] as our acoustic model. Wav2Vec 2.0 is a speech model for self-supervised learning of speech representations that masks the speech input in the latent space and solves a contrastive task defined over a quantization of the jointly learned latent representations.It takes a float array corresponding to the raw waveform of the speech signal. And since it was trained using connectionist temporal classification (CTC) so the model output has to be decoded using the Wav2Vec2CTCTokenizer. And for language model we used satistical base language model. Statistical language models, in essence, are the type of models that assign probabilities to the sequences of words. In this competition, we have used 5-gram Language model.

The starting point of our training was the Wav2Vec2 pre-trained weight from the best wav2vec2 model avaibale which was published in the competition of Bengali.Ai Speech Recoginition [6] [10]. We have trained around 17 epochs and got around 74% wer on our validation data. We didn't get satisfactory result from our wav2vec2 model training.

2) Whisper (Medium) Training: In second part of our training we used Whisper model as the base model. Whisper is a pre-trained model for automatic speech recognition (ASR) and speech translation published by OpenAI. The base model was trained on on 680k hours of labelled data, Whisper models demonstrate a strong ability to generalise to many datasets and domains without the need for fine-tuning. Whisper based model till now is the best model for general bengali automatic speech recognition (BnASR) task. The Whisper [8]

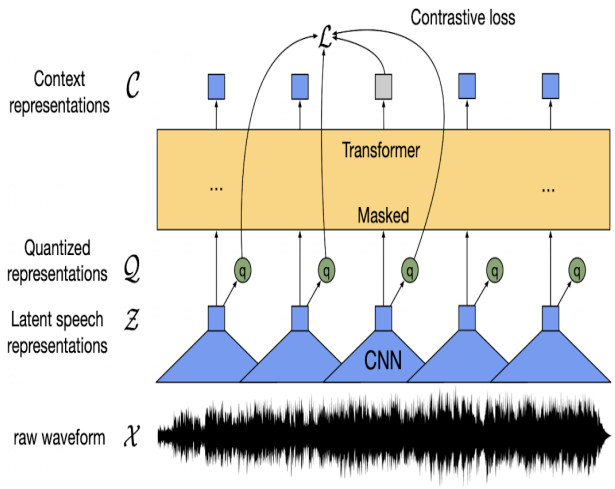

Fig. 2: Wav2vec2 model architecture

architecture is a simple end-to-end approach, implemented as an encoder-decoder Transformer. Input audio is split into 30-second chunks, converted into a log-Mel spectrogram, and then passed into an encoder. A decoder is trained to predict the corresponding text caption, intermixed with special tokens that direct the single model to perform tasks such as language identification, phrase-level timestamps, multilingual speech transcription, and to-English speech translation. We

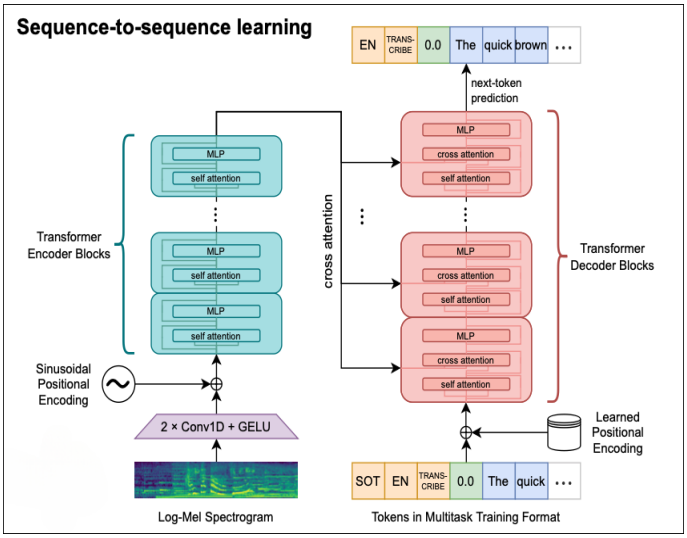

Fig. 3: Whisper model architecture

take whishper medium as our base model for the task. For the task we started training from the existing best model for BnASR [11]. We have trained around 7 epochs where we found the convergence of the model for the training dataset. This model gives use the best wer performance in the leaderboard.

3) Parameter Efficient Fine-tuning of Whisper: Then we explored weather we can improve the model performance of whisper model training more epochs and using other training schemas. As the competition time was short we trained whisper model using LoRA [9] architecture for faster training. We started the training from the same weights as the base whisper medium model that was used previously. We trained around 13 epochs with whisper medium model with LoRA. Parameters of lora training are given bellow.

TABLE I: LoRA Training Parameters

Parameter Name	Value
r	32
lora_alpha	64
target_modules	”q_proj”, ”v_proj”
lora_dropout	0.05
bias	none

The result of these experiments are given in the result section.

E. Inference

Automatic Speech Recognition is a vastly explored process now-a-days. Big companies and research organizations keep investing in this field because of the promise it holds. The huge level of research in this particular system has resulted in an improved latency of the whole ASR system with new innovations. The inference of our system was done by both the ’Insanely Fast Whisper’ system that uses flash attention 2 and ’Faster Whisper’ that uses the C transformer. These systems were used to improve the latency of the generation of text from speech data. The Wav2Vec2 inference was done in normal manner without any attention to the latency or accuracy since the model failed to learn as much as Whisper did.

*1) **Insanely Fast Whisper**:* Insanely Fast Whisper [12] method uses flash attention 2 [13] for its inference on specific GPUs. The primary inferences were done using this mention by intalling flash-attn package from pypi and then turning the Flash Attention parameter to true. This attention mechanism can result in an improved latency where an audio of an hour can take only 15 seconds to be transcribed. The Whisper inference pipeline by itself uses chunks of 30s and batches them for inference, but since the audio files were mostly 15 to 25 seconds long, it didn’t necessarily improve score much. The float16 quantization also improved the score by a good margin.

*2) **Faster Whisper**:* While flash attention 2 mechanism can result in insanely fast speed in inference, the overall technique might degrade the score a little bit. Our trained model for whisper improved score by 1.5% margin while inferencing without flash attention compared to the insanely fast whisper method. The Faster Whisper is another technique that converts a transformer based model to C language and then uses it to inference. The underlying mechanism, using C and float16 quantization improves latency by a good margin while keeping overall performance intact.

*3) **Post-processing**:* The post-processing of this part is very interesting. Mainly because of the uninterpretable token in the dataset, both training and test. As mentioned before, the

dataset contains '<>' token the most compared to other words, which means, upon wrong prediction of the uninterpretable tokens, the score can obviously be hurt by a good margin. Here, we took help of **word level timestamp**. The whisper models can already predict a word level timestamp of each predicted words. But, those timestamps are not always perfect. In [14], the author proposes a new method, to use Whisper as well as Wav2Vec2 model to predict the word level alignment of timestamps accurately. This method was groundbreaking and very much accurate. We used that method and used our Whisper model with a Wav2Vec2 model to explore our data more and figure out the duration between each '<>' token in the prediction string. We then thresholded the duration and for shorter thresholds, we removed the uninterpretable token. This technique resulted in a further 1% increase in the overall test set score.

III. Result

In this section we present our experiments results. As we described earlier we experimented with different model and different training architecture during training period. We used word error rate (wer) as our training metric. Overall experiments results are given in table II. Both of the public and private leaderboard result are recorded in the result table for better understanding of our models robustness.

TABLE II: Result analysis of different experiments

Model Name	Public Leaderboard (WER)	Private Leaderboard (WER)
wav2vec2	0.83911	0.84807
whisper medium	0.68950	0.70052
whisper medium with postprocessing	**0.68388**	**0.67536**
whisper medium with LoRA	0.69682	0.71129

IV. Conclusion

In this paper, we propose and efficient method for fine-tuning ASR models and inference with efficiency. Our methods and results helped us to achieve the 2nd postition in both of the public and private leaderboards with decent WER for both. Our efficient system uses only two deep learning models, one for transcribing the speech data and then incorporate itself with Wav2Vec2 model to get word level timestamps of the audio input for further analysis and post processing. Our proposed method does not use any denoiser models which are very impractical for daily use and might result in a costlier GPU for inference method.

References

[1] R. Ardila, M. Branson, K. Davis, M. Henretty, M. Kohler, J. Meyer, R. Morais, L. Saunders, F. M. Tyers, and G. Weber, "Common voice: A massively-multilingual speech corpus," *Proceedings of the 12th Conference on Language Resources and Evaluation (LREC 2020)*, pp. 4211–4215, 2022.

[2] A. Baevski, Y. Zhou, A. Mohamed, and M. Auli, "wav2vec 2.0: A framework for self-supervised learning of speech representations," *Advances in neural information processing systems*, vol. 33, pp. 12 449–12 460, 2020.

[3] A. Graves, S. Fernández, F. Gomez, and J. Schmidhuber, "Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks," in *Proceedings of the 23rd international conference on Machine learning*, 2006, pp. 369–376.

[4] H. Shahgir, K. S. Sayeed, and T. A. Zaman, "Applying wav2vec2 for speech recognition on bengali common voices dataset," *arXiv preprint arXiv:2209.06581*, 2022.

[5] S. H. Ujjal, A. M. Kabir, and M. A. Haque, "mtova: A multilingual task oriented virtual assistant for human computer communication," in *2023 IEEE International Conference on Telecommunications and Photonics (ICTP)*. IEEE, 2023, pp. 01–05.

[6] A. C. R. H. S. T. Addison Howard, Ahmed Imtiaz Humayun, "Bengali.ai speech recognition," 2023. [Online]. Available: https://kaggle.com/competitions/bengaliai-speech

[7] B. AI, "Bengali ai competition leaderboard," Online, 2023. [Online]. Available: https://www.kaggle.com/competitions/bengaliai-speech/leaderboard

[8] A. Radford, J. W. Kim, T. Xu, G. Brockman, C. McLeavey, and I. Sutskever, "Robust speech recognition via large-scale weak supervision," in *International Conference on Machine Learning*. PMLR, 2023, pp. 28 492–28 518.

[9] E. J. Hu, Y. Shen, P. Wallis, Z. Allen-Zhu, Y. Li, S. Wang, L. Wang, and W. Chen, "Lora: Low-rank adaptation of large language models," *arXiv preprint arXiv:2106.09685*, 2021.

[10] S. Place, "Bengali ai competition second place," Online, 2023. [Online]. Available: https://www.kaggle.com/datasets/qdv206/wv-shru-v3-s6

[11] Tugstugi, "Bengali ai competition first place," Online, 2023. [Online]. Available: https://www.kaggle.com/datasets/tugstugi/bengali-ai-asr-submission

[12] V. Srivastav, "insanely-fast-whisper," Online, 2024. [Online]. Available: https://github.com/Vaibhavs10/insanely-fast-whisper

[13] T. Dao, D. Fu, S. Ermon, A. Rudra, and C. Ré, "Flashattention: Fast and memory-efficient exact attention with io-awareness," *Advances in Neural Information Processing Systems*, vol. 35, pp. 16 344–16 359, 2022.

[14] M. Bain, J. Huh, T. Han, and A. Zisserman, "Whisperx: Time-accurate speech transcription of long-form audio," *arXiv preprint arXiv:2303.00747*, 2023.