Anadolu Psikiyatri Dergisi || ISSN: 1302-6631 || Volume 26; Number 4
https://doi.org/10.5281/zenodo.16258256

Dialect-based automatic speech recognition in Tamil

Saranya S, Sreeja K, Bharathi B

Department of Computer Science and Engineering,

Sri Sivasubramaniya Nadar College of Engineering, Chennai, 603110

Abstract

Tamil is a classical language that has a very ancient history and a distant and sumptuous literary grammar system. In any language, the way the people of a certain region express themselves is described as a dialect. Dialects are of great significance in the analysis of a language especially in the dialect identification systems mainly because most language dialects are extremely similar to each other. However, dialects cannot even be regarded as distinct and separate entities since they are tied to regional and often cultural characteristics. A survey of the available speech databases revealed that today there is no standard speech database collected for the Tamil dialects in the field of speech processing. Thus, to fill in the gap for the proposed system, efforts will be directed on creating a speech corpus that will facilitate dialect recognition for Tamil. This paper focuses on three main dialects of the Tamil language: Southern, Northern, and Western. In the present study, the process of transcribing spoken Tamil into written Tamil involves identifying words based on a fine-tuned model of the OpenAI Whisper model trained on the developed corpus. Notes: In this study, the Word Error Rate (WER) is applied to measure the performance of the developed system.

Keywords

Fine-tuning, Gaussian Mixture model, MFCC, OpenAI, Transfer learning, Tamil dialect speech corpus, Word error rate

1. INTRODUCTION

Since the beginning of time, speech has been the primary means of communication and information exchange. During performance of a speech recognition system, dialects of a language can be an issue because they are used for different purposes in different automatic speech recognition systems. Dialect is the manner in which a group of people, who happen to belong to the same geographical region pronounce language. Despite their many similarities, there are frequently significant variations at a number of language levels, including phonology, grammar, orthography, and separate vocabulary. Speaker's dialect differentiation will influence automatic speech recognition (ASR) in cases with forms of the same word being pronounced differently by speakers, familiar with a set of dialects. Dialect information will be in speech at different segmental levels. In this work, speech-based speech recognition for Tamil is explored. Spectral features are widely used in other speech processing methods.

2. RELATED WORK

An easy way to comply with APD paper formatting requirements is to use this document as a template and simply type your text into it. In this section, the related work on speech recognition is explored. [1] The authors built a dataset of various Telugu dialects for this purpose. To distinguish the dialects, they employed MFCC and its variants, such as delta and double delta MFCC coefficients. 39 feature vectors are taken from each frame of the utterance and assessed using GMM and HMM models to put the work into practice. It was found that GMM performs better than HMM model and that some words with comparable acoustic properties are not distinguished. In [2] The author built a dialect speech database consisting of spontaneous speech spoken by male and female speakers, for each dialect the duration of speech will be about 1-1.5 hrs. Prosodic aspects are represented by syllable lengths, pitch, and energy contours, whereas spectral features are represented by Mel frequency cepstral co-efficients(MFCC).

In their study [3], the authors implemented a speech recognition using the Mozilla DeepSpeech which is a relatively light on-demand revisiting of words that people speak. The models used for training were trained from the dataset of common voice available online as a non-restricted resource and via accessible computational clusters. The ASR model was tested and its best performance resulted in a WER of 24.7 % which inferior (higher) to Google's speech-to-text service.

ASR systems have been built for South Indian languages including Tamil and Kannada languages. The Tamil language has 65 phonemes and 38 syllabic units that is represented by using the subword-based proposal [4]. For Tamil ASR, 152 hours of training data and 65 hours of testing data have used to get the WER of 6.24%. For Kannada the details of the results achieved are, WER of 6.63% was obtained using 275 hours of training data and 72 hours of test data.

CNN based acoustic model was introduced for ASR in [5]. Due to this, the CNN-based model learns phonetic phoneme to speech signal mapping. In this work, the first and second derivatives of Mel-frequency features were employed as feature extraction which include Mel Frequency Spectral Coefficients (MFSC) and Gammatone Frequency Cepstral Coefficients (GFCC) that have been proved to improve the outcome of the model. The work proposed uses an open source, high quality, multi-speaker speech dataset and the training and testing accuracy achieved was 90.63% and 81.25% respectively. A couple of authors [6] tested three popular pre-trained models namely the XLSR Wav2Vec2 which is developed by Facebook and fine-tuned it with the Common Voice dataset. Another method of feature extraction, described in [7], is based on the Perceptual Linear Predictive (PLP) coefficients and Mel Frequency Cepstral Coefficients (MFCC).

To investigate the methods for CKSR (continuous Kannada speech recognition) In [8] used monophone, triphone, DNN-HMM, and GMM-HMM models and used the Kaldi toolkit. The MFCC technique was used to extract the feature vectors of the upcoming audio clips. The Kannada speech corpus was comprised 2800 subjects (1680 male, 1120 female) between 8 and 80 years old. The WER for the developed DNN-HMM and GMM-HMM speech recognition models were 8.23%, 5.23%, 4.05 and 4.64% as presented in table 5 and table 6. To the best of our knowledge, the only Tamil ASR work which focuses on continuous speech recognition is in [9], where HMM was presented as a suitable approach for low resource languages such as Tamil.

In [10], the authors introduced a new framework of Tamil speech word recognition which is composed of three phases. To begin with, Mel Frequency Cepstral Coefficients (MFCC) was obtained from set of training and set of testing samples. These were then utilized in training and testing of the Feed-Forward Backpropagation Neural Network (FFBNN). The results of this study indicated that this technique was effective compared to the traditional techniques such as Hidden Markov Models (HMM) and associative Artificial Neural Networks (ANN).

In the same manner, a speaker independent recognition system for Tamil words was suggested in [11], using Discrete Wavelet Transform (DWT) in association with multilayer perceptron network, where it was trained using the back propagation algorithm. The feature extraction implies that db4 wavelet was applied. From the eight levels in which the speech samples were decomposed, classification of these samples were done using first and second level approximation and detail coefficients.

This work is based on realizations from other dialect-based speech recognition systems in other languages and developing a Tamil dialect speech corpus to fine tune the Whisper model to enable it to recognize region-based speech utterances.

3. MOTIVATION

Speech-based dialect recognition is a difficult task due to factors like insufficient dialect-based speech corpus, subtle regional boundaries and variations in languages. Tamil language is one of the oldest languages in the world, as its literature corpus is enormous, and its grammar tradition is well-defined. Hence, the proposed system will enable translation which will help people from different regions of TamilNadu to easily interact with different applications. The system also provides easy access to information to all users irrespective of factors like age or education.

4. CHALLENGES

In this work, the Tamil dialects taken into account are Southern dialects spoken in Southern regions of Tamil Nadu such as Kanyakumari, Tirunelveli, Thoothukudi and Madurai. Western dialect spoken in

Coimbatore and Northern dialect spoken in Chennai. For Tamil dialects, there is currently no standard speech corpus available that can be used for speech processing research. The unavailability of dialect-based speech corpus is a reason for less research in dialect-based speech recognition system is not built for Tamil.

5. PROBLEM STATEMENT

To identify dialects and recognise speech utterances from different Tamil dialects.

Objectives

- To create a speech corpus for three different dialects of Tamil Nadu such as Northern Dialect, Southern Dialect and Western Dialect.
- To extract Mel frequency cepstral coefficients from the speech utterances.
- To fine-tune Open AI whisper model to recognise the Tamil speech utterances.
- To analyse the performance of the model using metrics like Word Error Rate (WER).

6. PROPOSED METHODOLOGY

In this work, speech utterances of different dialects are collected for corpus creation. It is separated as training data and testing data. The Open AI model is fine tuned for speech recognition task. The speech data and its corresponding transcript is separated as training data and testing data. For fine tuning Open AI model, Whisper Feature extractor is used to extract features from speech data and a tokenizer is used to create label ids. A data collator is used to handle input features and labels separately and the model is trained. The performance of the model is analyzed using word error rate. The steps involved in the proposed dialect speech recognition system is shown in Fig. 1 and MFCC feature extraction steps are shown in Fig. 2.

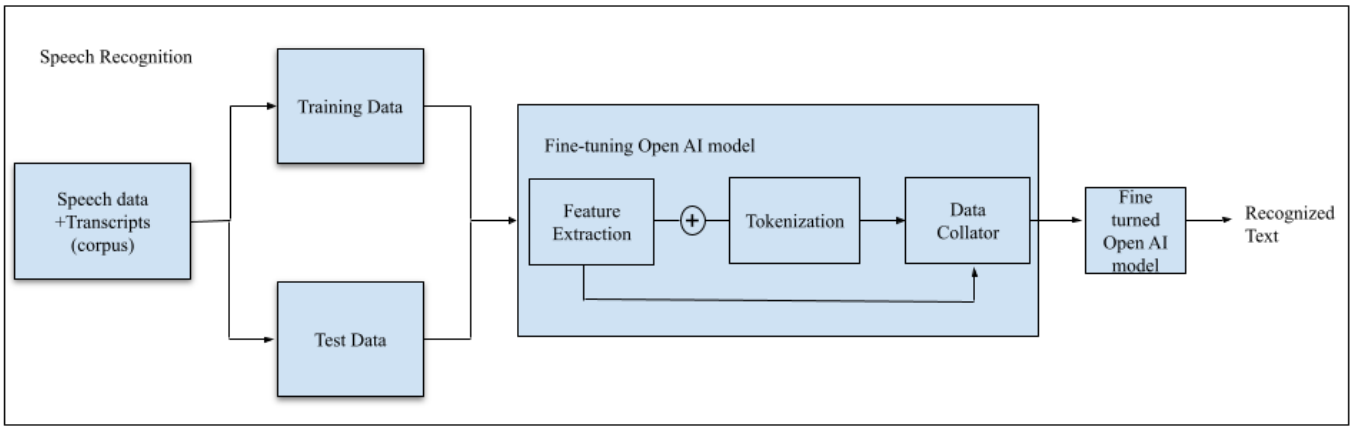

Fig. 1 Architecture diagram of proposed dialect-based dialect automatic speech recognition system

Mel frequency Cepstral Coefficient is a common and efficient spectral characteristic in a variety of speech-processing applications [10,12,13]. The MFCCs of a speech signal describes the entire shape of the spectral envelope. In contrast to standard cepstrum, which uses frequency bands for linear representation,The Mel Frequency Cepstrum (MFC) employs frequency bandwidths, which are arranged based on the Mel scale, an approximation of the human auditory system's auditory perception of sound. Eq. 1 defines the relationship between the frequency (in Hz) and Mel frequency.

$$\text{Mel}(f) = 1125 \ln(1 + f/700) \quad (1)$$

The pre-processing of the speech samples consists of the removal of the background noise that is accomplished by setting up a frequency threshold. Any of its components, which are below this threshold, regarded noise are deleted. To limit spectral leakages in the voice signal, the Hamming window method

divides the signal into windows. Because speech is quasi-natural in that it varies continuously, the speech signal is framed in the form of 20–40 ms of speech. We must frame the signal in order to obtain useful attributes or clues about the dialects. This signal frames 20 to 40 ms of speech, which is a relatively brief amount of time, and it gives useful information about the necessary dialects. Less than this size may not convey the crucial indications to distinguish the dialects. In order to obtain an accurate spectrum estimate, the frames must be separated properly. FFT (Fast Fourier Transform) is applied to the speech signal. There is a need to convert the signal from time domain into frequency domain. Signal conversion occurs There are key techniques to transform information to carry out the change from the time domain to the frequency domain. Multiply with a triangular band pass filter to get a smooth magnitude spectrum [1]. 20 band pass filters are used in this work. To determine the energy at each location, the sum of energies should be calculated. As humans are unable to perceive audio at higher frequencies, the channel is normalized using the Log function. Discrete cosine transform is used for each filter bank to get rid of filter bank overlaps. 20 feature vectors are ultimately collected from each frame.

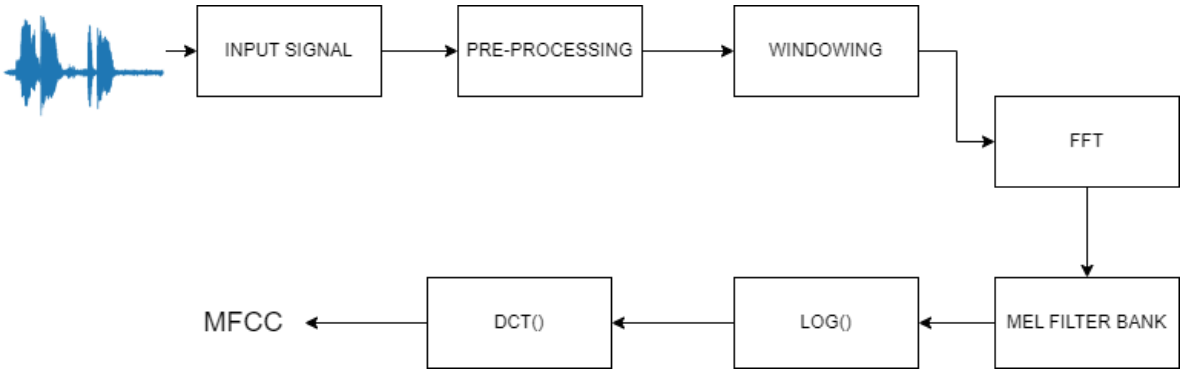

Fig. 2 MFCC features extraction steps

A. Data Creation

Although there is no standardized speech corpus for Tamil dialects, speech utterances are collected from different regions of Tamil Nadu. Sakthivel made a survey on Tamil dialects, based on the survey carried out by the department of Linguistics, Annamalai University, repeats the four regions of Kamil Zvelebil. The four major dialects covering the major regions of Tamil Nadu are,

- Northern Dialect - Madras, Chengalpet and North Arcot Districts.
- Central Dialect - Tiruchirapalli, Thanjavur and South Arcot Districts.
- Western Dialect - Salem, Dharmapuri, Coimbatore and Nilgiri Districts.
- Southern Dialect - Madurai, Ramnad, Tirunelveli and Kanyakumari Districts.

TABLE 1
Number of speech utterances collected from each dialect region

Dialects	Number of speech utterances
Southern dialect	428
Western dialect	281
Northern dialect	874

For this study, speech utterances are collected from Western, Northern and Southern regions of Tamil Nadu. The age group of speakers considered for this task is between 17 to 40. The speech corpus consists of both male and female speakers. The speech utterances were collected from various sources like Whatsapp, Youtube and through smart recorder. The speech utterances were in different formats like .mp3, .aac, .ogg etc. Those audio files were converted into .wav form using audio.online converter tool. The speakers who took part in the speech recording spoke about their own topics in their regional dialects. Thus, the speech corpus is created. Text transcription is created for these speech utterances manually. Number of speech utterances collected for each dialect are tabulated in Table 1.

B. *Speech Recognition System*

The next part of the proposed system is to recognise the speech. In the proposed system, OpenAI whisper model is fine-tuned using the speech utterances collected for this research work.

1) OpenAI Whisper:

The recognition of the speech is the next step in the suggested system. The speech utterances gathered for this study are used to fine-tune the OpenAI whisper model in the suggested system. Over 680,000 hours of supervised, multilingual, and multifunctional online data were used to train the Whisper ASR model. It is implemented using an encoder-decoder architecture. 30-second segments of audio input are transformed into a log-Mel spectrogram. The decoder predicts text captions. Some special tokens have been introduced in this sense to make the downstream task easier; the identification of the language used, the phrase-level time-stamping, multilingual speech transcription, and speech translation to English. The architectural diagram of the Open AI Whisper model is given in Fig. 3.

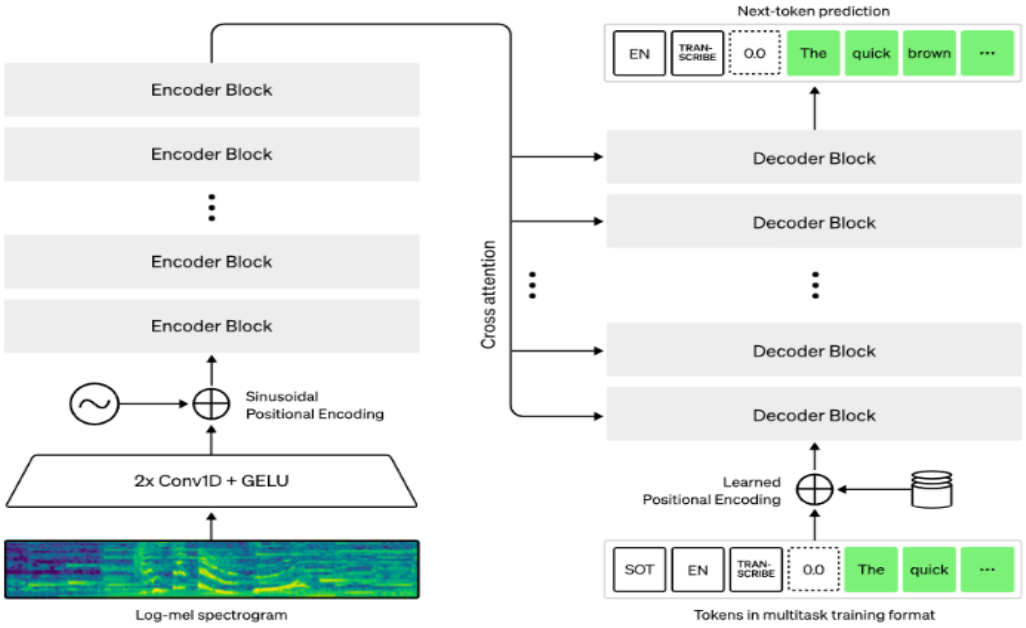

Fig. 3. Architecture of OpenAI Whisper

The input speech signal is resampled to 16 kHz and an 80-channel log-magnitude Mel spectrogram is determined with 25-ms windows and a stride of 10 ms. For feature normalization, the spectrogram values are globally normalized to lie between -1 and 1 with approximately zero mean over the pre-training data set. The input is processed by the encoder through a small stem comprised of two convolutional layers with kernel size 3, and GELU activation. The stride for the second convolutional layer for the temporal down-sampling is two. Sinusoidal positional embeddings are then added to the output of the convolutional stem and then a line of Transformer encoder blocks. These pre-activation residual connections are included in these Transformer blocks, and finally, the layer normalization is applied to the encoder's output. The learnt positional embeddings and shared input-output token embeddings are used by the decoder. The encoder and the decoder have the same width and the same number of Transformer blocks.

C. Fine-Tuning OpenAI Whisper

The steps involved in fine-tuning OpenAI Whisper model is given as follows:

- Data creation
- Data loading and processing
- Preparing Data
- Training and Evaluation
- Defining Data Collator
- Defining Training Arguments

1) Data Creation:

Data preparation is the first stage of every model training process. A csv file is created for the dataset where first column contains the audio file path and second column contains the transcripts of the respective audio files which is shown in Fig. 4.

2) Data loading and processing:

The second step will be data loading and processing the speech signal. The prerequisite for this model for the speech will be as follows:

- *Speech signal length: not more than 30 sec*
- *Speech file format: wav*
- *Sampling rate: 16000*
- *Type: mono*

	audio	sentence
2	/content/	பாத்தியோ என்னய
3	/content/	செரி செரி அப்போ ரெண்டு வார்த்தை பேசு பாப்போம் ஸ்பானிஷ் ல
4	/content/	இவ யம்பிளா இந்த சமயத்துல என்ன கூப்பிட்ற
5	/content/	இவ என்னத்துக்கு இப்போ விளிச்சியா
6	/content/	கேர்ள் நீ டாபிக் அனுப்புனா தான் நா அனுப்புவே நீ இப்படி ஜென்ரலா சொன்னா எனக்கு தெரியாது நீ கரெக்டா முதல்ல அனுப்பின மாதிரி டாபிக் டாப
7	/content/	இது யாருக்குள்ளது அந்த செகண்ட் பிக்சர் மேல கட்டாயிருக்கு ஃபுல்லா அனுப்பு
8	/content/	எபே இது கொவின் பேப்பர் மாதிரியே இல்ல டைப் பண்ண மாரியும் இல்ல இத எப்பிடி நம்ப
9	/content/	ஹாஸ்பிடல் பெய்ட்டு மிஸ் கிட்ட போன் அடிச்சி டக்குனு எக்ஸாம் எளுத போயிட்டேன்
10	/content/	நீ வராம போயிட்டா அப்டி இப்டி சொல்லிட்டு கொஞ்சம் எதாது சொல்லிட்டிருக்கு அதா கடுப்பாகுது
11	/content/	நாளைக்கு ஒரு நாள்தான் நிம்மதி ஒரு ஒன்பது மணி பத்து மணி வரைக்கும் தூங்கலாம்னு பாத்தா அவரு எட்டு மணிக்கு யளு டு எட்டுணு வச்சிருக்கா
12	/content/	காலேஜ் அட்மிஷன் இந்த லோகல் காலெஜெல்லா ஈஸியா கெடச்சிரும்
13	/content/	படிச்சு நல்ல மார்க் இருந்தா கெடைக்கும் இல்லனா கெடைக்காது
14	/content/	ஈஸியா போய் ஒரு சீட் எடுத்திர்கலாம் ஆனா வந்து கொஞ்சம் நல்ல காலேஜ்னா கொஞ்சம் நல்லா படிக்கணும் பிள்ளே
15	/content/	அதுக்கப்புறம் அந்த சீட்டுக்கான காசு அடைச்சு அந்த இத வந்து டவுன்லோட் பண்ணி எடுக்கணும் அந்த ரெசிப்ட்ட அப்டின்னு தா நெனைக்கே இது க

Fig. 4. Sample of Data created for fine-tuning OpenAI Whisper model

3) Feature Extractor, Tokenizer and Data preparation: The ASR pipeline can be broken into the three parts:

- *Whisper Feature Extractor*
- *Whisper Tokenizer*
- *Whisper Processor*

Whisper Feature Extractor

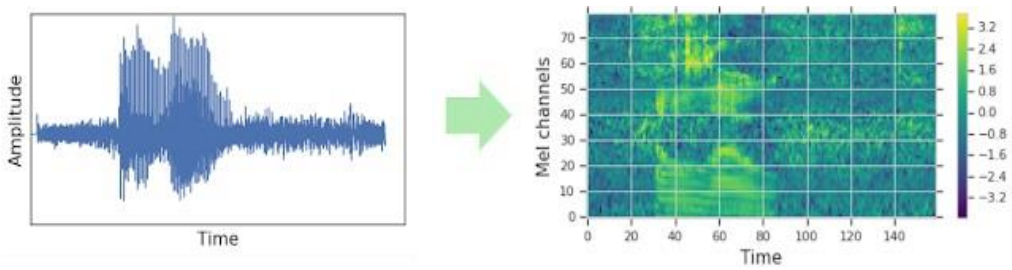

Fig. 5. Conversion of sampled audio array to log-Mel spectrogram

The model requires ready audio data through the effective processing capabilities of the Whisper feature extractor. The conversion process establishes consistency because speech signals with various lengths are transformed into fixed-length log-Mel spectrograms that maintain vital time-frequency information. The standardized input structure enables model training by creating a favourable condition that focuses on learning meaningful speech patterns. Speech signals become visible through the

structured waveform display of frequency patterns evolving in time which the log-Mel spectrogram creates as shown in Fig 5. The feature extractor establishes vital connections that transform original waveforms into a format suitable for the model architecture which enables effective speech recognition

Whisper Tokenizer

Text tokens that reflect the anticipated text's index inside the dictionary of vocabulary items are produced by the Whisper model. A series of text tokens are mapped to the real text string by the tokenizer. The transcriptions of the 96 pre-training languages are used to pre-train the Whisper tokenizer. As such, it possesses a large byte-pair suitable for nearly all multilingual ASR applications. The tokenizer adds "special tokens" to the beginning and ending of the sequence when encoding the transcriptions. These tokens include the language token, the task token, and the start/end of the transcript tokens. Special tokens are skipped during the label id decoding process, enabling the return of a string in its original input format.

Whisper Processor

The whisper feature extractor and whisper tokenizer are wrapped within one class called WhisperProcessor. This processor object inherits from WhisperFeatureExtractor and WhisperProcessor and can be used on audio inputs as well as model predictions when needed as required.

4) Training and Evaluation:

A pre-trained checkpoint must be loaded and properly set up for training. The Whisper model generates (forced decoder ids) which are token ids that are required as model outputs. Such token ids control the transcription language and task for zero-shot ASR. Word error rate (WER) metric will be used to measure the effectiveness of the model.

5) Defining Data Collator:

What makes a data collator of a sequence-to-sequence speech model unique is that it deals with input features and labels separately; the feature extractor has to process input features, the tokenizer, labels. With the feature extractor's pad method with return tensors=pt, the input features will be padded to 30 seconds, transformed into a log-Mel spectrogram of fixed dimension, and then converted to batched PyTorch tensors. Since the inputs are of fixed dimension and the input features can be transformed to PyTorch tensors, no extra padding is used in this case. But there is no padding on the labels. The sequences can be padded to the batch's maximum length using the tokenizer's pad technique. After that, the padding tokens are set to 100 to eliminate them from the loss computation. Following the removal of the transcript token's commencement, the label sequence is subsequently added later on in the training process.

6) Defining Training Arguments: The training arguments used here are as follows:

***output-dir:** It is a local directory where the weights of the model will be saved. This will be the repository name.*

***generation-max-length:** The limit of tokens to autoregressive generate throughout assessment.*

***save-steps:** Intermediate checkpoints during training can be saved.*

***eval-steps:** During training, evaluation of intermediate checkpoints will be performed based on every eval-steps training steps.*

***Trainer.train()** is used to begin training. Training will take ten to fifteen hours, depending on the GPU and the number of stages provided. The hyperparameters, such as learning rate, batch size, epochs, etc., are varied during the experiments. The whole ASR pipeline is handled by a transformer pipeline, which handles everything from preprocessing the audio inputs to decoding the model predictions. Gradio is used to create the demo. The Speech is recorded using the computer's microphone, fed into our fine-tuned Whisper model to transcribe the relevant text.*

7. EXPERIMENTAL RESULTS

***A.** Speech recognition by fine tuning OpenAI Whisper*

Fine tuning the model involves the following steps.

- Data preparation
- Data loading and processing
- Evaluation

B. Data Preparation

Audio	Sentence
/content/drive/MyDrive/SPEECH CORPUS/SOUTH/wav/trainingdata/South_F_01.wav	பாத்தியோ என்னய
/content/drive/MyDrive/SPEECH CORPUS/SOUTH/wav/trainingdata/South_F_02.wav	செரி செரி அப்போ ரெண்டு வார்த்தை பேசு பாப்போம் ஸ்பானிஷ் ல
/content/drive/MyDrive/SPEECH CORPUS/SOUTH/wav/trainingdata/South_F_03.wav	இவ யம்பிளா இந்த சமயத்துல என்ன கூப்பிட்ற
/content/drive/MyDrive/SPEECH CORPUS/SOUTH/wav/trainingdata/South_F_04.wav	இவ என்னத்துக்கு இப்போ விளிச்சியா
/content/drive/MyDrive/SPEECH CORPUS/SOUTH/wav/trainingdata/South_F_05.wav	கேர்ள் நீ டாபிக் அனுப்புனா தான் நா அனுப்புவே நீ இப்படி ஜென்ரலா சொன்னா எனக்கு தெரியாது நீ கரெக்டா முதல்ல அனுப்பின மாதிரி டாபிக் டாபிகா அனுப்பு அப்போ நான் வந்து பேசி அனுப்புறேன்

Fig. 6. Sample sentences in the speech corpus

A csv file is created for the dataset where the first column contains the audio file path and the second column contains the transcripts of the respective audio files. The sample file for the data preparation is shown in Fig. 6.

C. Data loading and processing

Whisper feature extractor and whisper tokenizer are combined to create the processor which is shown in Fig. 7.

```
WhisperProcessor:
- feature_extractor: WhisperFeatureExtractor {
  "chunk_length": 30,
  "feature_extractor_type": "WhisperFeatureExtractor",
  "feature_size": 80,
  "hop_length": 160,
  "n_fft": 400,
  "n_samples": 480000,
  "nb_max_frames": 3000,
  "padding_side": "right",
  "padding_value": 0.0,
  "processor_class": "WhisperProcessor",
  "return_attention_mask": false,
  "sampling_rate": 16000
}
```

Fig. 7. Whisper Processor

A demo is built for the speech recognition system. Through microphone of our computer, we can record the speech and input it into our fine-tuned Whisper model to transcribe the text as represented in Fig. 8.

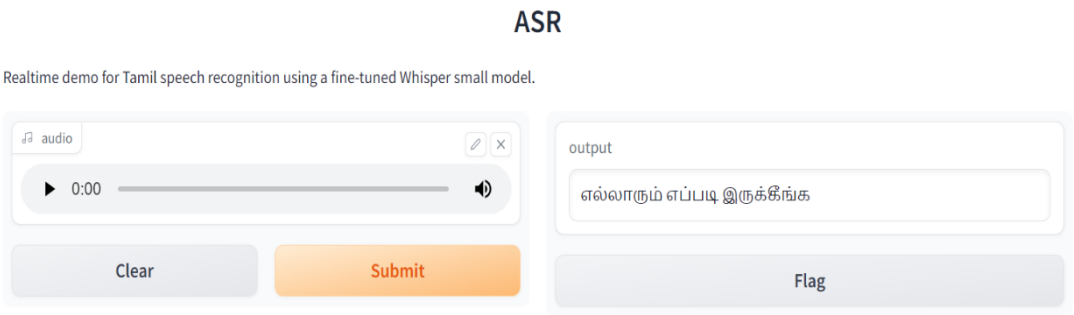

Fig. 8. Live demo of Tamil speech recognition using gradio

Step	Training Loss	Validation Loss	Wer
500	0.001100	2.328347	350.485437
1000	0.004600	2.257277	217.475728
1500	0.000900	2.366130	232.038835
2000	0.000900	2.420424	200.970874
2500	0.000800	2.457450	207.766990
3000	0.000800	2.486677	202.912621
3500	0.000800	2.509395	205.825243
4000	0.000800	2.527006	203.883495

Fig. 9 Word Error rate of Base model

First, the performance of the speech recognition system is tuned with various multilingual whisper models. The WER using the Base model with learning rate = 1e-5, batch size = 32, and epochs = 4000 is illustrated in Fig. 9.

Using small model WER with, learning rate = 1e-5, Batch size = 32 and Epochs = 250 is given in Fig. 10.

The WER using Tiny Model with Learning rate= 1e-5, Batch size= 32, and Epochs = 1500 is depicted in Fig. 11.

Step	Training Loss	Validation Loss	Wer
50	No log	0.273560	45.569620
100	No log	0.284335	45.569620
150	No log	0.289326	45.569620
200	No log	0.292177	44.303797
250	No log	0.294624	44.303797

Fig.10 Word Error Rate of Tiny Model

Step	Training Loss	Validation Loss	Wer
500	0.596600	1.860955	160.194175
1000	0.001900	2.148136	164.077670
1500	0.001100	2.280927	165.048544

Fig. 11 Word Error rate of small mode

From the fig 9 - 11, it has been noted that WER of small model is better than base and tiny model. The small model is fine-tuned by changing various hyperparameters and the result is shown in Table 2.

From Table 2, it has been noted that WER is 44.3% when the learning rate as *1e-7* and batch size as *6*. From these results it has been noted that performance is improved when the Open AI whisper small model is fine-tuned using the custom-created speech corpus. The performance of speech recognition is further increased by increasing the training samples in each dialect.

TABLE 2
WORD ERROR RATE USING SMALL MODEL BY VARYING THE HYPER PARAMETERS

EPOCHS	LEARNING RATE	BATCH SIZE	WER (in %)
250	le-5	16	100
25	le-6	8	84
125	le-5	8	76
250	le-7	6	44.3

8. CONCLUSION

The proposed research is related to the recognition of dialect-based speech utterances in Tamil. For that purpose, a custom speech corpus was created, which included three major Tamil dialects. An ASR system was then developed by fine-tuning OpenAI Whisper with data from these dialects' training sets. Experimental results demonstrate that the quality of Whisper' fine-tuned model is more sensitive on the number of training utterances for each dialect, which further confirms the assertion that more data variety and quantity are required to achieve the improved accuracy of the dialect specific ASR.

9. CONFLICT OF INTEREST

The authors declare that they do not have any conflict of interest.

10. DATA AVAILABILITY

Data will be made available on request.

REFERENCES

[1]. S. Shivaprasad and M. Sadanandam, Dialect recognition from telugu speech utterances using spectral and prosodic features, International Journal of Speech Technology, 2021, pp. 1–10.
[2]. K. S. Rao, S. Nandy, and S. G. Koolagudi, Identification of hindi dialects using speech, WMSCI-2010, 2010.
[3]. M. H. Changrampadi, A. Shahina, M. B. Narayanan, and A. N. Khan, End-to-end speech recognition of tamil language. Intelligent Automation & Soft Computing, Vol. 32, no. 2, 2022.
[4]. B. Pilar et al., Subword dictionary learning and segmentation techniques for automatic speech recognition in tamil and kannada, arXiv preprint arXiv:2207.13331, 2022.
[5]. S. Girirajan and A. Pandian, Convolutional neural network based automatic speech recognition for tamil language, 2022, pp. 91–103.
[6]. D. Srinivasan, B. Bharathi, T. Durairaj et al., Ssncse nlp@ lt-edi-acl2022: Speech recognition for vulnerable individuals in tamil using pre-trained xlsr models, in Proceedings of the Second Workshop on Language Technology for Equality, Diversity and Inclusion, 2022, pp. 317–320.
[7]. Kumar and V. Mittal, Hindi speech recognition in noisy environment using hybrid technique, International Journal of Information Technology, Vol. 13, 2021, pp. 483–492.
[8]. P. Kumar and H. Jayanna, Development of speaker-independent automatic speech recognition system for kannada language, Indian J Sci Technol, Vol. 15, 2022, pp.333–342.
[9]. V. Radha, C. Vimala, and M. Krishnaveni, Isolated word recognition system for tamil spoken language using back propagation neural network based on lpcc features, Computer Science & Engineering, Vol. 1, no. 4, 2011, p. 1.
[10]. S. Rojathai and M. Venkatesulu, A novel speech recognition system for tamil word recognition based on mfcc and ffbnn, European Journal of Scientific Research, Vol. 85, no. 4, 2012, pp. 578–590.
[11]. P. Sivaraj and M. Rama, Recognition of isolated spoken words using dwt, Int J Eng Sci Res, Vol. 2, no. 9, 2012, pp. 1187–1196.
[12]. R. Ranjan and A. Thakur, Analysis of feature extraction techniques for speech recognition system, International Journal of Innovative Technology and Exploring Engineering, Vol. 8, no. 7C2, 2019, pp. 197–200.
[13]. N. J. Ibrahim, M. Y. I. Idris, M. Yakub, Z. M. Yusoff, N. N. A. Rahman, and M. I. Dien, Robust feature extraction based on spectral and prosodic features for classical arabic accents recognition, Malaysian Journal of Computer Science, 2019, pp. 46–72.
[14]. D. S. Park, W. Chan, Y. Zhang, C.-C. Chiu, B. Zoph, E. D. Cubuk, and Q. V. Le, SpecAugment: A Simple Data Augmentation Method for Automatic Speech Recognition, in Proc. Interspeech 2019, 2019, pp. 2613–2617.