Experiments on Front-End Techniques and Segmentation Model for Robust Indian Language Speech Recognizer

Conference Paper · March 2014

DOI: 10.1109/NCC.2014.6811284

2 authors:

Murali Karthick Baskar
Brno University of Technology
7 PUBLICATIONS **3** CITATIONS

SEE PROFILE

Sriranjani Ramakrishnan
Indian Institute of Technology Madras
5 PUBLICATIONS **1** CITATION

SEE PROFILE

Experiments on Front-End Techniques and Segmentation Model for Robust Indian Language Speech Recognizer

Sriranjani R [1], *Murali Karthick B* [2] *and Umesh S* [2]
Department of Applied Mechanics [1], Department of Electrical Engineering [2]
Indian Institute of Technology Madras, Chennai 600036, India
Email: 1) am12s036@smail.iitm.ac.in 2) {ee13s010 and umeshs@ee.iitm.ac.in}

***Abstract*—Recent contributions in the area of Automatic Speech Recognition (ASR) for Indian Languages has been increased. This paper serves as a comprehensive study of different feature extraction methods namely MFCC, PLP, RASTA-PLP and PNCC. An attempt to find out which of these front end techniques performs better for real world Indian Language data is analyzed experimentally. Then, an isolated word recognizer is built for three Indian languages (i.e., Tamil, Assamese and Bengali) under real world conditions and investigates the importance of handling long silence using segmentation method. The experimental analysis shows that PNCC provides better performance for clean data whereas MFCC shows improved performance in case of multi-condition speech data.**

***Index Terms*—Speech recognition, Noise robustness, Feature extraction, Segmentation, Silence handling, real world speech, comparison of front-end techniques**

I. Introduction

Building a robust ASR for Indian languages is one of the active area of research. Some of the challenges faced to develop a robust recognizer are different dialects, pronunciations and environment conditions. Past studies shows the collection of various speech corpus in different conditions for Indian languages [1, 2]. In [3, 4], improved acoustic model is built for a specific Indian language like Hindi. Noise robust features for Indian languages were implemented using different algorithms in[5, 6]. In [7], acoustic models are built using each of the feature extraction methods namely MFCC, LPCC and PLP and merged with voting rule for improving performance. In all the above, the major focus was on improving the acoustic model using the existing or new features.

This motivates us to choose a robust feature extraction method among the existing methods for building a better Indian language ASR. The recognition accuracy obtained using different features are compared with the results of AURORA2 database [18]. The feature extraction methods analyzed in this paper are Mel Frequency Cepstral Coefficients (MFCC) [8], Perceptual Linear Prediction (PLP) [9] coefficients, Relative Spectra - PLP (RASTA-PLP)[10] and Power Normalized Cepstral Coefficients (PNCC) [11].

Isolated word recognizer is built for three Indian languages namely Tamil, Assamese and Bengali using different features. Mandi database used for our experiments contains data collected from the target population i.e. the farmers in their working environment. More detailed explanation on mandi database is given in Section IV.

978-1-4673-5952-8/13/$31.00 © 2013 IEEE

An attempt to improve the acoustic model for real world data using segmentation method has also been experimented in this paper. Segmentation method improves performance of real world systems by handling long silence effectively. Our experimental study shows that, MFCC works better for multi-condition data while PNCC performs well for clean speech. The performance improves further by using segmentation method along with the MFCC features.

The rest of the paper is organized as follows: Section II explains comparative study of different feature extraction methods. In Section III, we explain about the segmentation process and its importance. The difficulties involved in building a real world speech recognizer for Mandi database and its experimental setup are explained in Section IV. The results are analyzed in Section V. The conclusion of the paper and its extension are mentioned in Section VI.

II. Feature Extraction Methods

The main objective of feature extraction is to capture the important aspects of the speech signal irrespective of it being corrupted by other factors like noise, environmental degradation etc. In [12], the front end processing of the digital signal processing is segregated into 3 main parts: Spectral shaping, Spectral Analysis and parametric transform. The parameters are considered as the concise representation of the signal. Conversion of analog signal to digital and filtering process will be done in spectral shaping. In the spectral analysis section, the sampled signal is divided into frames and its spectrum is processed. This process will form the feature vectors from every processed frame. The parametric transform will do the post processing like Normalization, dynamic feature extraction etc. over the feature vectors.

A. Initial processing

The sampled input speech signal is converted into frames by using the short time Fourier analysis of the signal using the window. There is significant overlap between each window shift in order to prevent the discontinuities along the edges. The most successful feature extraction process is the one which include the psychophysical auditory processing. Cepstrum / Spectrum analysis is then carried out in order to extract the feature vectors.

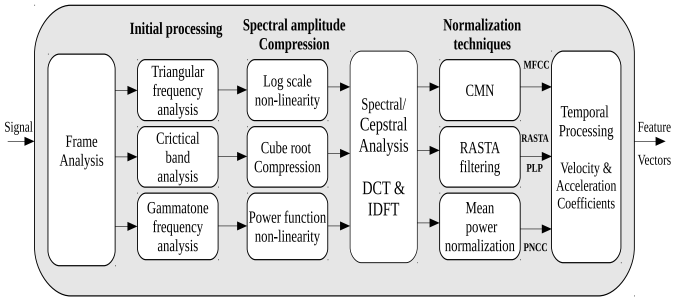

Figure 1. General framework of MFCC, PLP/RASTA-PLP and PNCC

The MFCC is the most common feature extraction method used in speech processing. It uses the Mel-filter bank which models the hair spacings of the ear along the basilar membrane. PLP is based on Linear prediction and concentrated on auditory processing. PNCC was motivated by a desire to get robust features comparable to MFCC and PLP in computational complexity , strongly influenced by attributes of auditory processing. Figure 1 explains the general framework of all the feature extraction methods. More detailed comparison [13] is given below:

- Frame analysis: All the above feature extraction methods, obtain the short time power spectrum using the Fourier transform of the windowed speech frame. The signal is windowed typically in the range 20-30 ms with an window overlap of 10 ms
- Filter bank analysis: MFCC uses triangular Mel filter banks based on Mel scale. Bark scale is used for PLP which provide the critical band analysis. While PNCC uses the Gamma tone filter bank which are linearly spaced in Equivalent Rectangular bandwidth in the range 200 - 8000Hz

- Pre-emphasis and Loudness compensation: To compensate for unequal sensitivity in the hearing, pre-emphasis is done. MFCC and PNCC uses a first order high pass filter $H(z) = 1 - \alpha z^{-1}$. PLP uses equal loudness function given by the equation $E(\omega)$, where ω is the frequency in rad/sec

$$E(\omega) = \frac{\left(\omega^2 + 56.8 * 10^6\right)\omega^4}{\left(\omega^2 + 6.3 * 10^6\right)^2\left(\omega^2 + 0.38 * 10^9\right)} \quad (1)$$

- Spectral amplitude compression: MFCC uses log scale compression. PLP uses a cube root (1/3) amplitude compression of the loudness equalized critical band spectral estimate. A power function non-linearity with exponent of 1/15 for dynamic range compression is used in PNCC.
- All-pole modeling and cepstral analysis: MFCC analysis computes cepstral coefficients from the log Mel-filter bank using a discrete cosine transform (DCT). PLP analysis gives Linear Prediction (LP) coefficients through Inverse discrete Fourier transform (IDFT). This is post processed to form cepstral coefficients. In PNCC, mean power normalized coefficients are passed through DCT to form cepstral coefficients

B. Feature normalization

Front-end processing techniques include normalization of the computed static feature vectors. Sensitivity to changes in long term is reduced by average log power spectrum using the following methods:

- RASTA filtering: Each frequency band in the short term spectrum is band pass filtered with a filter which has spectral zero at zero frequency. Thus the spectral estimate becomes less sensitive to slowly varying components of speech. This helps in reducing both noise distortion and channel effects. This RASTA filtering is done in PLP feature extraction method to form RASTA-PLP features.
- Cepstral mean normalization (CMN): The mean of each coefficient across frames is subtracted from the original cepstral vectors. It is done to reduce channel offset along with stationary speech components. The shift of the C_0 coefficient due to logarithmic nonlinearity in MFCC is removed using CMN.
- Mean power normalization: Medium time non-linear processing is done in PNCC to suppress the effects of additive noise and room reverberation by a non-linear series of operations along with temporal masking using analysis windows in the order of 50 - 150 ms. To minimize the effects of power law non-linearity, mean power normalization is carried out by dividing the incoming power by a running average of the overall power.

C. Temporal processing

Temporal processing is done to capture the coarticulation effects and the temporal information. Difference between the cepstral components are calculated by first applying weighted summation to the time sequence of cepstral vectors. Time derivative of the cepstral coefficients for each frame is taken. Velocity and acceleration coefficients are taken by finding the simple difference and double difference between adjacent frames.

III. Segmentation Method

Segmentation is the process of classifying phonemes, syllables, words etc., An initial estimation of the boundaries of phonetic segments is an important stage in ASR. This initial estimation is obtained using the following forced alignment method.

Given a HMM m_i for each phonetic segment $(i = 1, 2, ..., N)$ of the utterance, the observations $\mathcal{O}_1, \mathcal{O}_2, ..., \mathcal{O}_T$ of the utterance to be segmented are mapped against the HMM sequence, which are created from the phonetic transcription of the sentence. The Viterbi algorithm [14] is applied to perform forced alignment as it finds the best path on the network. The backtracking of the path gives the initial segmentation. For each segment of the utterance, a model m_i $(i = 1, 2, ..., N)$ is estimated. The joint probability of the utterance given the models of all phonetic units in the utterance is computed and taken as reference $P_{ref} = p(\mathcal{O}_1, \mathcal{O}_2, ..., \mathcal{O}_T/m_1, m_2, ..., m_N)$. Then for each segment boundary b_j $(j = 1, 2, ..., N - 1)$, the hypotheses of moving the boundary one frame to the left or one frame to the right is analyzed. For each movement of the boundary to the left or right, the join probability

$$P_{left} = p(\mathcal{O}_1, \mathcal{O}_2, ..., \mathcal{O}_T/m_1, m_2, .., \hat{m}_j^{left}, \hat{m}_{j+1}^{left}, .., m_N) \tag{2}$$

or

$$P_{right} = p(\mathcal{O}_1, \mathcal{O}_2, ..., \mathcal{O}_T/m_1, m_2, .., \hat{m}_j^{right}, \hat{m}_{j+1}^{right} ..m_N) \tag{3}$$

is computed respectively. $\hat{m}_j^{left}$ and $\hat{m}_j^{right}$ represent left and right estimated models respectively. If the value higher than the reference i.e. $max \{P_{ref}, P_{left}, P_{right}\}$ is P_{left}, then the boundary is moved to the left and P_{ref} is actualized as P_{left} and vice-versa. The algorithm is iterated until no movement produces an increase on the joint probability.

In our work, segmentation method is exploited by finding the silence segments which occurs before and after each utterance. Voice activity detection (VAD) is used efficiently only if the data is clean (i.e. only silence and speech portions). Since we build model for real time data, the utterance has noise along with the silence, hence the segmentation method is preferred over other methods like VAD. Here we refer to the model built using this method as Segmentation model.

Procedure in building segmentation model is as follows:

- Speech, noise and silence portions of each utterance is found using forced alignment
- Initial HMM is built by specifying the speech portions and it is called as flat-start HMM. In flat-start HMM, silence portions are given less priority compared to speech portions. Since the initial model built is without long silence, every phonetic segment is mapped to a HMM state with very less indulgence of silence
- Re-estimation of the initial model is done
- Triphone model is built using the re-estimated model

IV. Experimental setup

Tamil, Bengali and Assamese corpus are obtained from the Mandi database. HTK Toolkit [15] was used for model building. Experimental set up includes feature extraction for all the 3 languages and building segmentation models for the same.

A. MANDI database

The speech data was collected for the spoken dialog system for getting the price information of the Agricultural commodities in various districts across India. The data was collected for 6 Indian languages namely Tamil, Hindi, Telugu, Marathi, Bengali and Assamese. This Government project is being sponsored by the Department of Information Technology (DIT) .Out of these 6 languages, we chose Tamil, Bengali and Assamese for our work.

The data was collected from the end user (farmers) in the real time field environment. The data collection volunteers will travel to the site and collect data over the telephone by encouraging the farmers to speak and answer to the spoken dialog system. Both the genders equally participated while collecting data. The volunteers also collected metadata information from the farmers. The real time environment recorded during data collection includes, clean , noisy, background noise, vehicle horn, background music etc. These environmental changes are captured as noise fillers during acoustic model building in the transcription. The common fillers include “background noise”, “background speech”, “vehicle horn”, “cry”, “laugh” etc. Table II shows the number of fillers and phones for each of the language. The data has also many regional and social dialects which are captured by giving alternate pronunciations to each word in the dictionary for each language.

B. Feature extraction experiments

Features are extracted using all the 4 methods namely, MFCC, PLP, RASTA-PLP and PNCC for all the 3 Indian languages as shown in II. MFCC and PLP are implemented using HTK software toolkit [15]. PNCC is implemented using the open source matlab code available in [16] while RASTA-PLP is implemented from [17].

Each of the feature extraction process involves various configuration parameters. Window length of 25ms and a frame shift of 10ms is kept constant for all these methods. 39 dimension feature vectors for each of the methods are extracted using the following configuration parameters:

- *MFCC, RASTA-PLP and PNCC:* In MFCC, RASTA-PLP 13 cepstral coefficients and 13 power normalized cepstral coefficients in PNCC are concatenated with velocity and acceleration coefficients
- *PLP:* 12 cepstral coefficients concatenated with energy, velocity and acceleration coefficients are used with a LPC order of 12.

C. Segmentation model experiments

Acoustic model building for real time data, includes a baseline model and segmentation model for each of the Indian Languages. Model building is done for each of the feature extraction methods. To handle real world noise, transcription of each utterance included fillers. Separate modeling of noise fillers is done. The model was trained with real world data and also tested with the same. Table I includes the details about number of hours and number of words (vocabulary) used in testing as well as training for each language.

Details on number of Gaussian mixtures and tied states for each language is mentioned in Table II.

Table I
SIZE OF VOCABULARY (VOCAB) AND NUMBER OF HOURS OF DATA (D-DISTRICTS, C-COMMODITIES) USED FOR TRAINING AND TESTING

Language	Training		Testing			
	# Hours	Vocab Size	# Hours		Vocab Size	
			D	C	D	C
Assamese	35.93	296	1.25	2.28	27	108
Bengali	28.4	306	3.42		306	
Tamil	44.6	8170	4.15	4.28	35	305

Table II
NUMBER OF TIED-STATES, GAUSSIAN MIXTURES (GM), PHONES AND FILLER

Language	# Tied-states	# GM	Phones	Fillers
Assamese	1108	16	38	1
Bengali	566	16	61	39
Tamil	552	16	51	14

Training: HMM model with 3 emitting states was build for all phonetic units. Ergodic 5 state HMM model was built to handle silence. Noise fillers was modeled as 5 state HMM by tying all 3 emitting states of each noise filler into single state. Flat start HMM model is build initially using the whole training data. Force alignment is done for each utterance to segment the whole word into phoneme segments. This alignment information is used to build a monophone model for each phonetic unit. Baum-Welch re-estimation of models is done for 4 iterations. 16 mixture monophone model is built by incrementing the mixture component in multiples of 2 after every set of re-estimation. The final triphone model built is used as baseline model.

Steps involved in building segmentation model:

- Baseline model is used as reference model to perform segmentation mentioned in Section III for every utterance
- The identified boundary portions of speech are specified as frames along with each utterance to build a flat-start HMM. For example: Starting frame of speech $-($ 20 frames to ending frame of speech $+($ 20 frames$)$ is mentioned as boundary information. Here, we approximately use 20 frames before and after each speech portion in every utterance to adjust the offset.
- Further training is done to create a triphone model for the segmentation method

Testing: Tamil and Assamese contains separate test data for commodities and districts. Bengali test data combined both commodities and districts into a single test set. Noise fillers are allowed in language model to handle real world test data. The language model is created using word-net. Some of the methods used in modeling the word-net is as follows:

- It contains the information of each word instance and word-to-word transition.
- The noise in test data is handled by allowing noise fillers in word net

Scoring: System performance is tested by calculating word accuracy. The algorithm to find the accuracy aligns the output of the system to the reference transcription. This algorithm is an optimal string match based on dynamic programming [18]. Once the optimal alignment has been found, the number of substitution errors (S), deletion errors (D) and insertion errors (I) can be calculated.

$$\%\,Accuracy = \frac{N - D - S - I}{N} \times 100 \quad (4)$$

V. RESULTS AND DISCUSSION

Comparison of various feature extraction methods for Mandi data is shown in table III. From the results, it is observed on an average that, MFCC performs better than other feature extraction techniques for Indian Languages. The noise robust feature extraction techniques like PLP, RASTA-PLP (RPLP) and PNCC fails to outperform in real world environment. On an average, the amount of noise in each language is represented as: Noise in Tamil data $>$ noise in Bengali data $>$ noise in Assamese data. Even though the noise robust feature extraction methods like RASTA processing and PLP fails in this real world scenario, MFCC and PNCC are almost comparable for most of the conditions. It is observed that, PNCC results outperform MFCC for Assamese commodities data. This is due to the fact that the noise in Assamese data is less compared to other languages. The comparison of recognition performance for each feature extraction method is observed as: MFCC $>$ PNCC $>$ RASTA-PLP $>$ PLP. We tried to verify the observed pattern of results with AURORA2 database. This standard database is chosen mainly because: 1) The utterances contains isolated digits similar to the test set of Mandi database which has isolated districts and commodities. 2) Data is available under various noisy level (clean, SNR 20 etc.) and noisy environments (babble, car, subway etc.). More detailed description about the AURORA2 database is mentioned in [18].

Table III
% RECOGNITION ACCURACY USING DIFFERENT FEATURE EXTRACTION METHODS FOR ALL LANGUAGES

% Accuracy	MFCC	PLP	RPLP	PNCC
Assamese - Districts	**94.22**	92.8	94.13	93.87
Assamese - Commodities	78.29	75.98	79.74	**83.66**
Bengali	**90.95**	86.31	89.85	90.94
Tamil - Districts	**82.37**	74.04	77.16	80.29
Tamil - Commodities	**76.88**	66.98	68.53	70.24

Similar pattern of results was observed from experiments on AURORA2 database as shown in Fig 2 and Fig 3.

Figure 2 shows the results for various feature extraction methods with model trained on clean speech and tested under various noise conditions. Figure 3 compares the results for various feature extraction methods with model trained on multi-condition speech and tested under various noise conditions. The advantage of training on clean data is to handle any type of noise without distortions. This is well suited to represent all the speech information, but this model will not be robust to handle any distortions in the signal. Training on multi-condition data include the distortions from the speech signals, hence these models will be robust for any given environment with noise.

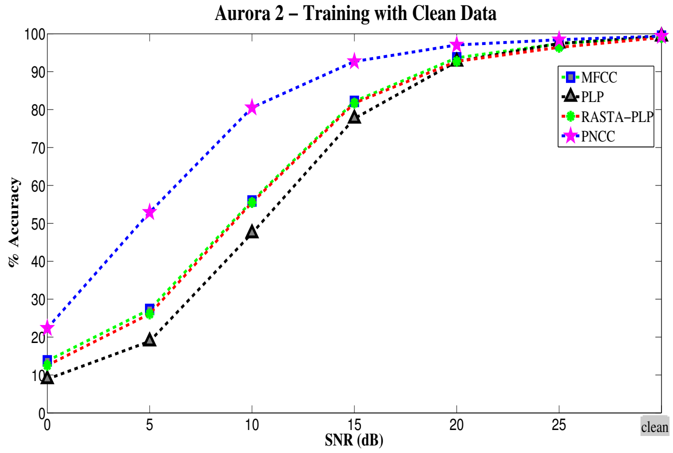

Figure 2. Comparison of Recognition Accuracy for model trained with Aurora2 Data containing Clean Speech and tested under various noise levels (SNR-5 to clean)

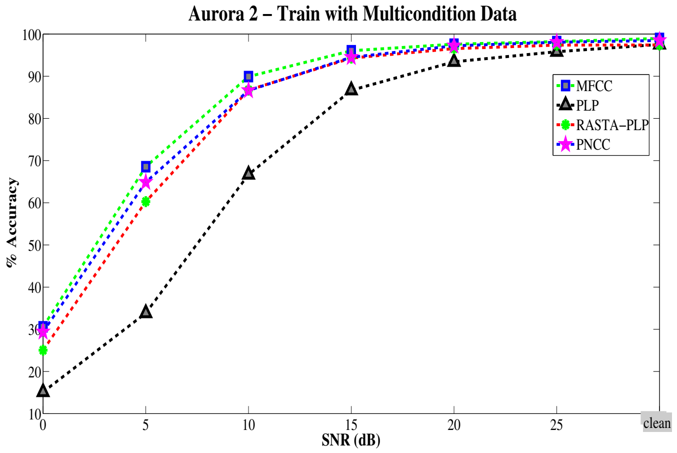

Figure 3. Comparison of Recognition Accuracy for model trained with Aurora2 Data containing Multicondition Speech and tested under various noise levels (SNR-5 to clean)

From Fig 2, we can observe that, for models build using clean data, PNCC features works better compared to all the other methods. This pattern is similar to the results obtained for Assamese data which is less noisy. For multi-condition training as shown in Fig 3, MFCC works better compared to other features. Our real time Indian Language data is similar to multi-condition data, hence the same pattern of results are observed for Tamil (Districts and Commodities), Assamese Districts and Bengali which are noisier than Assamese Commodities.

The results for the segmentation method is mentioned in table IV. It is observed that, segmentation method gives improved performance compared to the baseline model which was built using the MFCC features.

For example, consider an utterance containing word "Ellu" in Tamil. In baseline model, this utterance gets wrongly recognized as "Nellu" whereas the segmentation model correctly recognizes it as "Ellu". The noise portions which have not seen during training, corrupts the triphone $sil-e+ll$ model with noise. Since, $sil-e+ll$ is trained only for segmented speech portions, the triphone $sil-e+ll$ is robust to handle any test condition. Spectrogram of noisy and clean test utterance containing "Ellu" is shown in Fig 4 and Fig 5 respectively.

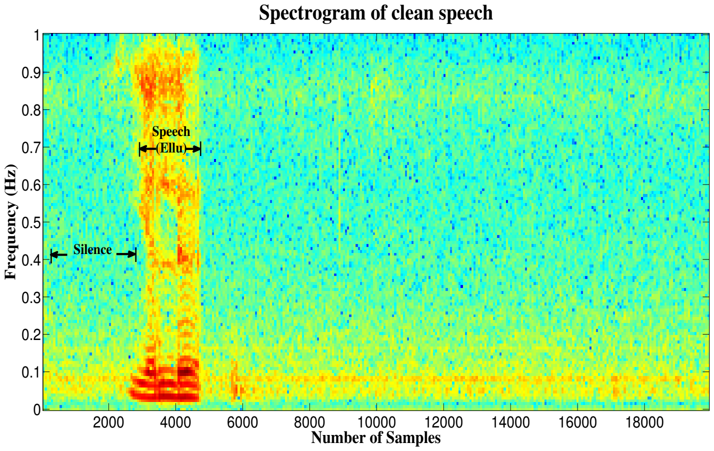

Figure 4. Spectrogram of Word "Ellu" in Clean Condition (Long Silence)

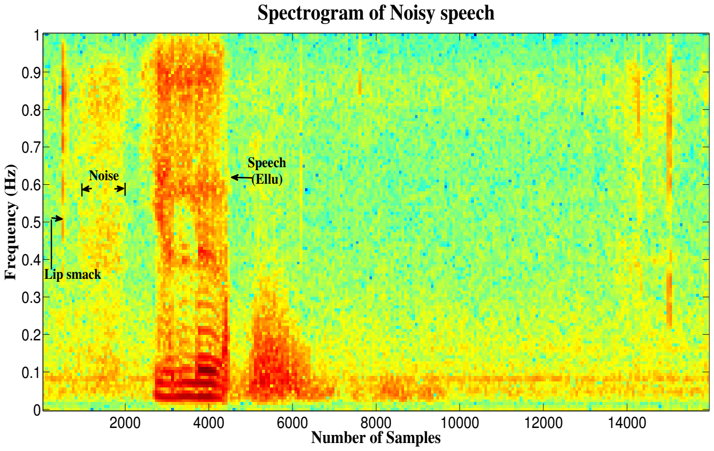

Figure 5. Spectrogram of Word "Ellu" in Noisy Condition

Table IV
% Recognition Accuracy using Segmentation model for all languages

% Accuracy	Baseline	Segmentation model
Assamese - Districts	94.22	**94.4**
Assamese - Commodities	78.29	**83.35**
Bengali	90.95	**91.02**
Tamil - Districts	82.37	**83.74**
Tamil - Commodities	76.88	**78.33**

Percentage Accuracy of commodities increases significantly compared to districts for each language, since the vocabulary size of districts is very less compared to commodities. The results shown here are for models built using MFCC features. But the same pattern of improvement holds for models built with other feature extraction methods. We have chosen MFCC, since it performs better on average for Indian languages as shown in Section IV.

VI. Conclusion and Future Work

In this paper, an experimental study of different feature extraction methods are performed for Indian Languages. A comparative analysis is done for all the feature extraction methods using a three stage system namely, initial processing, feature normalization techniques and temporal processing. Even though there are differences in each of the methods, there exists lot of similarities to be put into a common framework. Our study shows that, MFCC performs better compared to other feature extraction methods for real time data of Indian Languages. It is also observed that, PNCC outperforms other methods for data trained on clean environment. Real world data of Indian languages is similar to multi-condition data in AURORA2 and hence the same pattern of results is observed.

It is also shown that, segmentation model improves performance for data with long silence. Even if the silence occurs along with noise, it gives comparable performance as the data with long silence. The results shown in this paper are for models built using MFCC features. The similar variation in performance is seen for models built using other feature extraction methods. This is a initial phase of work on Indian Language real time data to find optimal features and improve the performance using segmentation. Future work includes aligning the boundaries of speech segments precisely by improving the segmentation algorithm to handle real time data.

VII. Acknowledgement

This research was supported in part by the Technology Development in Indian language programme of the Ministry of Communication & Information Technology, Govt. of India. The DIT-ASR consortium includes IIT Madras, IIT Bombay, IIT Guwahati, IIT Kanpur, IIIT Hyderabad, Tata Institute of Fundamental Research Mumbai and Center for Development of Advanced Computing Kolkata. The authors are grateful to all the members of Consortium for collecting and transcribing agricultural Commodity data.

References

[1] S. Agarwal, K. Samudravijaya, and K. Arora, "Recent advances of speech database development activities for indian languages," in *International Symposium on Chinese Spoken Language Processing (ISCSLP 2006)*, 2006.

[2] G. Anumanchipalli, R. Chitturi, S. Joshi, R. Kumar, S. P. Singh, R. Sitaram, and S. Kishore, "Development of indian language speech databases for large vocabulary speech recognition systems," in *Proc. SPECOM*, 2005.

[3] P. Saini and P. Kaur, "Automatic speech recognition-a review," *International journal of Engineering Trends & Technology*, pp. 132–136.

[4] A. Thakur and R. Kumar, "Automatic speech recognition system for hindi utterances with regional indian accents: A review 1."

[5] S. S. D. R. Dev Amita, Aggarwal, "On the performance of front-ends for hindi speech recognition with degraded and normal speech," *Symposium on Translation Support Systems, STRANS 2001, IIT Kanpur*, 15.

[6] M. I. Vishal Chourasia, Samudravijaya K and M. Chandwani, "Hindi speech recognition under noisy conditions," *J. Acoust. Soc. India*, vol. 54.

[7] M. Kumar, R. K. Aggarwal, G. Leekha, and Y. Kumar, "Ensemble feature extraction modules for improved hindi speech recognition system," *International Journal of Computer Science*, vol. 9.

[8] S. B. Davis and P. Mermelstein, "Comparison of parametric representations for monosyllabic word recognition in continuously spoken sentences," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 28, no. 4, 1980.

[9] H. Hermansky, "Perceptual linear predictive (PLP) analysis of speech," *J. Acoust. Soc. Am.*, vol. 57, pp. 1738–52, Apr. 1990.

[10] H. Hermansky, N. Morgan, A. Bayya, and P. Kohn, "RASTA-PLP speech analysis," Tech. Rep. TR-91-069, International Computer Science Institute, Berkeley CA, 1991.

[11] C. Kim and R. M. Stern, "Power-normalized cepstral coefficients (pncc) for robust speech recognition.," in *ICASSP*, pp. 4101–4104, IEEE, 2012.

[12] J. Picone, "Signal modeling techniques in speech recognition," *Proceedings of the IEEE*, vol. 81, no. 9, pp. 1215–1247, 1993.

[13] B. Milner, "A comparison of front-end configurations for robust speech recognition.," in *ICASSP*, pp. 797–800, IEEE, 2002.

[14] A. Bonafonte, A. Nogueiras, and A. Rodriguez-Garrido, "Explicit segmentation of speech using gaussian models.," in *ICSLP*, ISCA, 1996.

[15] S. Young, "The htk hidden markov model toolkit: Design and philosophy," *Entropic Cambridge Research Laboratory, Ltd*, vol. 2, pp. 2–44, 1994.

[16] R. M. S. Chanwoo Kim, "PNCC in Matlab," 2012. [Online]. Available: http://www.cs.cmu.edu/ robust/archive/algorithms/PNCC_IEEETran.

[17] D. P. W. Ellis, "PLP and RASTA (and MFCC, and inversion) in Matlab," 2005. [Online]. Available: http://www.ee.columbia.edu/ dpwe/resources/matlab/rastamat/.

[18] D. Pearce and H.-G. Hirsch, "The aurora experimental framework for the performance evaluation of speech recognition systems under noisy conditions.," in *INTERSPEECH*, pp. 29–32, ISCA, 2000.