Short Term Context based Fast Multilingual Acoustic Model for Low Resource Languages

Bipasha Sen[1*], *Aditya Agarwal*[1*], *Mirishkar Sai Ganesh*[2], *Anil Kumar Vuppala*[2]

[1]Microsoft India
[2]International Institute of Information Technology, Hyderabad

bise@microsoft.com, agarwal.aditya@microsoft.com, mirishkar.ganesh@research.iiit.ac.in, anil.vuppala@iiit.ac.in

Abstract

Multilingual automatic speech recognition (ASR) systems have led to a major step forward towards building robust ASR systems for languages with low resource availability by increasing coverage for individual languages. State of the art multilingual systems are developed with sequential networks such as bidirectional long short term memory networks (BLSTMs) to capture long term temporal dependencies. Training and inference in such sequential models are computationally expensive which poses a significant challenge in terms of scalability and real-time applications. In this paper, we propose an alternate architecture based on short term contextual temporal features learned on convolutional neural networks (CNNs) with a non-sequential discriminative network. We use three low resource Indic languages, Gujarati, Tamil, and Telugu to ascertain that our proposed architecture trains 5.5× faster and reduces the inference time by a factor of 26 while maintaining comparable word error rates (WERs) against our baseline BLSTM network.
Index Terms: Multilingual ASR, Low Resource, CNN-DNN

1. Introduction

Multilingual automatic speech recognition (ASR) system is a single entity capable of transcribing speech utterances for multiple languages with a shared phone space. The data set for multiple languages are combined and a common phone set is built. The common phone set is a distinct set of combined individual phone sets. Multilingual ASR systems have seen tremendous growth and have proven to be a viable solution for building robust speech recognition systems often outperforming the monolingual counterparts, especially for languages with low resource [1–5].

Conventionally, an ASR system is a modular system comprising of multiple sequential components. Each of these components is optimized individually. The first component extracts state of the art handcrafted features such as Mel frequency cepstral coefficients (MFCCs), perceptual linear prediction (PLP) from raw speech signals. Additionally, forced alignment techniques are used to generate output phone labels for these extracted features. The second component, an acoustic model trained on the extracted feature - alignment label pairs, estimates the phone probabilities for given acoustic representations. The third component, word reconstruction, takes in the estimated phone probabilities as input and predicts a sequence of words using a language model.

Recently, there have been substantial developments on end to end (E2E) acoustic modeling which combines multiple components to optimize the system as a single unit. Techniques based on attention networks [6, 7], connectionist temporal classification (CTC) [8–10] transcribe acoustic frames into phones without the need of any predefined alignments. Techniques based on Convolutional neural networks (CNNs) transcribe raw speech utterances directly into phones, eliminating the step of hand-crafted feature extraction [11–16].

The discriminative networks in the acoustic models are largely built on sequential networks such as recurrent neural networks (RNNs), long short term memory (LSTM) networks [17–19]. Sequential networks have the inherent property of capturing long term contextual dependencies in a sequence. Fields like natural language processing, machine translation are largely dependent on such sequential architectures. Such networks are however computationally expensive due to limited scope in parallelization. Deep neural networks (DNNs) on the other hand is a non-sequential architecture, providing computational efficiency but at a cost of low WERs [12].

Phones have very short contextual dependencies. For instance, the pronunciation of phoneme *k* is dependent on its neighboring phonemes in cat (*k-**ae**-t*), car (*k-**aa**-r*), hack (*hh-**ae**-k*), sky (*s-k-**ay***). However, in the sentence *"this is a cat" (dh-ih-s ih-z **ah** k-**ae**-t)*, the pronunciation of phoneme *k* is dependent only on neighboring phones, ***ah*** and ***ae*** but is independent of all the other phonemes in the sentence [20]. CNNs have been researched in the field of monolingual ASR as a feature extraction layer to improve the WERs on DNNs while maintaining the computational efficiency by *learning* the short term contextual dependencies on the acoustic frames [12].

In this paper, we build upon our prior work based on a joint acoustic model for low resource languages [1]. We take inspiration from the work done on CNNs in [11] and [12] to learn short term contextual temporal features on acoustic frames using convolutional layers. We experiment with both end-to-end and conventional ASR approaches based on non-sequential CNN-DNN architecture to draw a comparative analysis between the two of them. We use three low resource Indic languages, Gujarati, Telugu, and Tamil, with a combined training dataset of 75 hours. We use WER as the evaluation metric for all of our experiments. Our proposed system speeds up the training time by 5.5× and inference time by 26× while maintaining comparable WERs with our baseline BLSTM network. We achieve further improvement in the training and inference time by a factor of 30× and 65× respectively at the cost of significant degradation in WERs.

The rest of the paper is organized as follows. Section 2 describes the various system components of our experiments. Section 3 describes the experimental setup. Section 4 presents results with a comprehensive and comparative analysis. Section 5 concludes our paper by stating the inferences drawn.

[*]Bipasha and Aditya contributed equally to this work.

2. System Components

2.1. Convolutional Neural Networks

The domain of images and videos often witness very high dimensional data commonly in the range of 10^6. Training deep neural networks on such high dimensional data generates noise while tremendous machine power is needed to process such large fully connected layers. Convolutional neural networks reduce dimensionality whilst retaining important spatial features such as contour boundaries, edges, simple curves, etc. The extracted low dimensional features are then fed to a discriminative model to perform classification tasks.

A convolutional neural network is made up of stacked convolutional layers where each convolutional layer is a combination of convolution, activation, and pooling.

Mathematically, a convolution operation is expressed as:

$$C[m, n] = (i * k)[m, n] \tag{1}$$

where * denotes the convolution operation. *C* is the output of the convolution operation, *m* and *n* are the convolution output indices, *k* is a filter, also known as kernel and *i* is multi-dimensional input.

A non-strided convolution operation is performed as:

$$(i * k)[m, n] = \sum_j \sum_l k[j, l]i[m + j, n + l] \tag{2}$$

where non-strided denotes that the sliding step for the filter is 1. *j* and *l* are the dimensions of the filter.

The output dimension of the non-strided convolution operation is given as

$$d(C) = [p - j + 1, q - l + 1] \tag{3}$$

where *p* and *q* are the input dimensions.

Speech signals, like images, are high dimensional data where just a 1 second long speech sample with a frame rate of 16000Hz consists of 16000 features. Each feature represents a sample in time. Speech signals can then be viewed as a 1-dimensional input vector with values representing a time-dependent variation.

Conventionally, generic hand-crafted features like MFCC are extracted to reduce the dimensionality of the speech signals and represent the time-variation as a fixed-dimension vector. For instance, MFCC is represented as a 39 dimensional feature vector. CNNs as a replacement of hand-crafted features have been extensively researched [11–15] to extract contextual temporal features.

In our experiments, we have used CNNs as an extraction layer on raw speech signals to capture the lower-dimensional representations. We have also used CNNs on hand-crafted MFCC features to capture higher dimensional representations as a means to reverse any loss during MFCC feature extraction.

2.2. Sampling Raw Signals

Raw input speech signals have variable lengths, whereas CNN expects inputs to be of fixed dimension. Hence, the raw signals are first sampled into feature vectors of fixed dimensions. The dimension of the sampled feature vector f is given by,

$$d(f) = s \times \frac{t}{1000} \tag{4}$$

where s is the sampling rate in Hz and t is the duration in ms. The raw speech utterances are sampled every l ms apart.

The speech signals in our dataset have a frame rate of 16kHz and are sampled every 10 ms over a window of 25ms.

2.3. Mel Frequency Cepstral Coefficients

Mel frequency cepstral coefficients are the most commonly extracted features for ASR systems. The features are extracted through a series of predefined transformation on the raw speech signals to capture core information [21]. The set of transformations includes pre-emphasis to boost the energy in high frequencies, windowing for sampling the audio, edge smoothing, discrete Fourier transformation (DFT), Mel scale transformation as an attempt to mimic human ear perception, discrete cosine transformation (DCT) to produce uncorrelated features.

In our experiment, MFCC features have been extracted using Kaldi [22] recipe. The MFCC features are sampled from the raw speech signals every 10 ms over a 25ms window.

2.4. Context Window

Sequential models like LSTMs, GRUs, and seq2seq models capture coarticulation across consecutive phones. Such models are inherently complex and computationally intensive. Moreover, such models are equipped to capture long term contextual dependencies wherein models capturing minimum duration short term contextual dependencies lead to similar or improved results [20].

CNNs can be trained to capture such short term contextual dependencies by providing context to the current frame. Context windows represent the frames on the left and the right of the current frame. Instead of passing just the current frame along with the phone label to the model, a window of length $(m + n + 1) \times f$ is passed, where *m* represents the number of context windows on the left, *n* is the number of context windows on the right and f is the frame length.

3. Experimental Setup

3.1. Data

The Data is a subset of Interspeech 2018's Low Resource Speech Recognition Challenge for Indian languages by Microsoft and SpeechOcean.com dataset [1] [23].

India is a country with more than 1500 recognized languages. Out of these, 30 languages have more than one million speakers and 22 languages have been accorded with the official status [24]. Such diversity in spoken languages poses a significant challenge in obtaining sizable training data to train robust monolingual systems for each of these languages. This dataset was released as an effort to explore robust multilingual systems as a means to overcome the challenge of data limitations.

Table 1: *#Utterances included in training, dev and test set*

Languages	Train Set	Dev Set	Test Set
Gujarati	18307	4500	3075
Telugu	12667	3200	3040
Tamil	15712	3900	3081

The data includes three Indic languages namely Gujarati, Tamil, and Telugu spoken by multiple speakers. The combined training data of all three languages is 75 hours. The test and the validation data is 5 hours per language. Text transcription along with the lexicon for the entire data is included in the dataset.

[1]The dataset is available at `https://msropendata.com/datasets/7230b4b1-912d-400e-be58-f84e0512985e`.

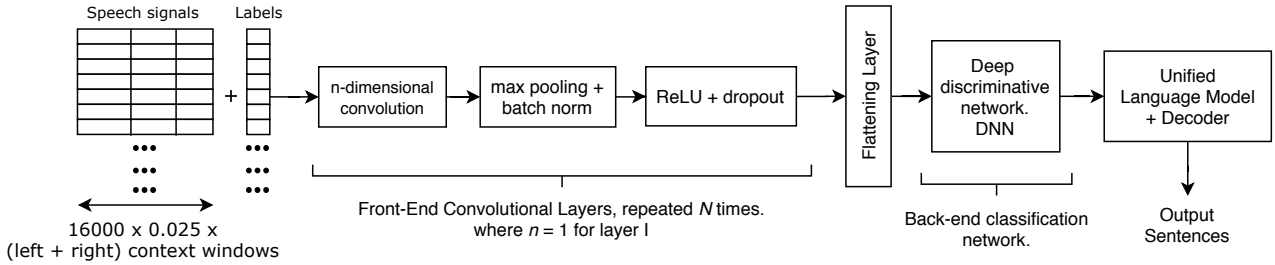

Figure 1: *Design of proposed CNN based architecture*

State of the art systems were built on the full dataset, which has 120 hours of trainable data. The baseline DNN based system in the challenge has a WER of 27.79, 34.97, and 25.47 on Gujarati, Telugu, and Tamil respectively [23].

A unified data set is created by combining the raw speech utterances, extracted MFCC features, phone set, and phoneme transcriptions of all three languages. This data set is used to train and validate all the acoustic models mentioned in this paper. The test set is language-dependent to obtain the WER for individual languages.

Unified Language Model is built on the combined corpus of all three languages using the SRILM toolkit mentioned in section 3.2. This language model has been used for all the experiments mentioned in this paper.

3.2. Toolkits

The Kaldi Speech recognition toolkit [22] has been used. SRILM toolkit was used for language modeling. MFCC feature extraction, modified KneserNey smoothed tri-gram models were built for forced alignments using the Wall Street Journal Kaldi recipe. LibriSpeech recipes were used to decode and score the system. Pytorch-Kaldi toolkit [25] was used for the development of acoustic models.

3.3. Baseline: Bidirectional Long Short Term Memory

The baseline model is a BLSTM network built on MFCC features with 3 hidden layers and 550 hidden units in each layer. tanh activation function is used with a 20% dropout in each layer. Layer normalization is used along with a batch size of 8. The learning rate is kept at 0.0016 with a halving factor of 0.5 with the improvement threshold of 0.001. RMSprop optimizer function is used. The MFCC features are generated using the techniques mentioned in section 2.2. Negative Loss Likelihood (NLL) is used as the loss function for the baseline and all the subsequent experiments mentioned in the paper. It is expressed as:

$$L(y) = -\sum_{i} log(y_i) \tag{5}$$

where y_i is the prediction for the i^{th} frame.

3.4. Proposed Architecture

Our CNN based acoustic system consists of two components, the front-end and the back-end. The front-end is a generative feature extraction network based on deep CNNs and the back-end is a discriminative network based on DNNs. The components are connected by a flattening layer and are trained and optimized as a single unit. Figure 1 represents the high-level design for the proposed end-to-end architecture. Each CNN layer performs convolutions, max pooling, batch normalization with rectified linear units (ReLU) activation. The DNNs consist of stacked fully connected layers with each layer made up of 1024 ReLUs.

The end-to-end system is trained on a feature vector - label pair where each feature vector is a concatenation of left context window, current acoustic frame, and right context window. The utterances are decoded with the decoder and a unified language model mentioned in section 3.2. The flattening layer flattens the multi-channel CNN output to a single-dimensional input to the discriminative network.

4. Experimental Results

This section presents a comprehensive and comparative analysis between different CNN based architectural configurations and the baseline BLSTM network.

Table 2 presents a comprehensive view of the experimental results obtained on models with different configurations. Table 3 presents a comparative analysis of computational time and average WER degradation of different configurations against the baseline model. The unit of the training time is kept variable to enable better readability. The inference time is calculated as T/N where T is the total inference time on the test set and N is the number of examples. All the models are trained and inferences are drawn on single-core NVIDIA TITAN Xp GPUs in a multi-core GPU setup.

Table 2: *WER of different models in %*

Models + Context	Gujarati	Telugu	Tamil
lstm + mfcc	**18.36**	**25.23**	**24.32**
cnn + raw + $\{0, 0\}$	24.06	31.23	30.96
cnn + raw + $\{-1, +1\}$	23.92	30.66	29.58
cnn + raw + $\{-2, +1\}$	**21.13**	**27.80**	**25.76**
cnn + raw + $\{-1, +2\}$	22.65	30.32	28.96
cnn + raw + $\{-2, +2\}$	23.87	32.09	31.20
cnn + mfcc	25.05	31.13	30.78

Two different CNN based architectures are trained. The first architecture is an end-to-end system trained on raw speech signals to enable the CNNs to *learn* sophisticated short term contextual features directly on raw data. The second architecture is based on the conventional ASR approach trained on handcrafted MFCC features.

Table 3: *Performance comparison on the average % WER, average % WER degration, training and inference time.*

				Training		Inference	
Exp	Models + Context	Avg. WER	Avg. WER deg.	Time	Speed Up	Time	Speed Up
1	lstm + mfcc	**22.63**	-	$\sim$ 4.5 days	-	780 ms	-
2	cnn + raw + $\{0, 0\}$	28.75	-6.12	7.84 hours	$\sim 13.5\times$	15ms	$\sim 52\times$
3	cnn + raw + $\{-1, +1\}$	28.05	-5.42	11.56 hours	$\sim 9\times$	29ms	$\sim 26\times$
4	cnn + raw + $\{-2, +1\}$	**24.89**	-2.26	19.08 hours	$\sim 5.6\times$	30ms	$\sim 26\times$
5	cnn + raw + $\{-1, +2\}$	27.31	-4.68	21.63 hours	$\sim 5\times$	32ms	$\sim 24\times$
6	cnn + raw + $\{-2, +2\}$	29.05	-6.42	24.42 hours	$\sim 4.4\times$	89ms	$\sim 8.7\times$
7	cnn + mfcc	28.98	-6.35	3.55 hours	$\sim 30\times$	12 ms	$\sim 65\times$

4.1. Deep Convolutional Layers on Raw Speech Signals

The front-end is a CNN composed of 3 hidden layers. The 1st, 2nd and 3rd layer has 8, 8 and 2 channels respectively. The length of the kernel in the first layer is 128. Kernel length is halved for each consecutive layer. Max pooling of length 5 is applied on the first layer and a length of 3 is applied on the other two layers. Drop out of 15%, 30% and 20% is employed for the 1st, 2nd and 3rd layer respectively. A learning rate of 0.0008 is employed with a halving factor of 0.5 and an improvement threshold of 0.001. ReLU activation function is employed at each layer.

The back-end DNN consists of 5 hidden layers. Each of the hidden layers is made up of 1024 units with a dropout of 10%. ReLU activation is applied to each layer with softmax activation on the output layer. A learning rate of 0.0004 is applied with a halving factor of 0.5 and an improvement threshold of 0.001. The end-to-end system is trained on mini-batches of 64 with batch normalization and RMSprop optimizer.

The architecture is trained on raw speech signals sampled to obtain 400 dimensional feature vectors using the technique mentioned in section 2.1. Different experiments are performed varying the size of the context window from 0 to 2 while keeping the CNN and DNN layers constant to find the ideal configuration. Adding 2 context windows on the left and right expands the 400 dimensional feature vector to $(2 + 2 + 1) \times 4 = 2000$ dimension.

Our best model achieves a WER of 21.13, 27.80, and 25.76 is on Gujarati, Telugu, and Tamil respectively with 2 left and 1 right context window (Exp 4, Table 3). The results are comparable to the baseline with an average degradation of only 2.26 WER. A computation boost of $5.6\times$ in training time and $26\times$ in inference time is obtained with this configuration taking only an average of 30ms to infer on an average of 5.85s long speech utterance compared to 780ms on the baseline.

It is observed from Table 3 that decreasing the size of the context window boosts up the training and inference computations but at a cost of degradation in the WERs. Models trained on acoustic frames with no context (Exp 2) and large context (Exp 6) observe a significant average degradation of $\sim$ 6.2WER. This indicates no context doesn't provide all the relevant articulation and coarticulation information for the phones while providing large contexts introduces more noise than relevancy. It is also observed that adding more context on the left (Exp 4) gives slightly better WER as compared to adding more context on the right (Exp 5) for the same context window size with comparable training and inference time speed up indicating that the left context contains more relevant information compared to the right context.

4.2. Deep Convolutional Layers on MFCC features

The front-end is a deep CNN composed of 3 hidden layers. The layers consist of 80, 60, 60 output channels respectively. The kernel length of 10, 3, 3 with max-pooling length of 3, 2, 1 is applied. The back-end is a deep neural network with 4 hidden layers, each layer composed of 1024 units. A dropout of 15% is applied in each layer of the end to end system with a learning rate of 0.08 and a halving factor of 0.5 with the improvement threshold of 0.001. ReLU activation is applied in every layer barring the output layer which employs softmax activation. The system is trained on a batch size of 128 with batch normalization and stochastic gradient descent optimizer.

The highest boost in the training and inference time is achieved from this configuration (Exp 7, Table 3) with an inference time of an average $12ms$ on an average 5.85s long speech utterance. This is a $65\times$ boost compared to $780ms$ inference time on the baseline. However, this comes with a significant degradation in the WER with an average degradation of 6.35. The WER performance is on par with the CNN architecture on raw speech signals with no context window with a speedup of 3ms on inference time.

5. Conclusion

In our previous work, we proposed a joint acoustic model based on HMM-SGMM and RNN-CTC for training a multilingual speech recognition system. In this paper, we investigated a non-sequential discriminative approach based on the features extracted by CNNs. Experimental results show that such systems vastly reduce the training and inference time while producing comparable WERs against our sequential baseline model based on BLSTM. This shows great potential and promise for CNN based models in real-time applications for achieving low latency and can also be used to quickly bootstrap multiple low resource multilingual ASR systems. We also experimented with different context window configurations and observed that a context window of size 2 on the left and 1 on the right produces the most optimal WERs. The best inference time was obtained on the model trained on MFCC features boosting the inference time by $65\times$. CNN based architectures can be further researched to improve and obtain better WERs compared to sequential models while speeding up the training and inference time.

6. Acknowledgements

We would like to thank Sunayana Sitaram, Rajeev Gupta and Sandipan Dandapat, who are researchers at Microsoft, for their valuable feedback and guidance.

7. References

[1] H. Krishna, K. Gurugubelli, V. V. R. V, and A. K. Vuppala, "An exploration towards joint acoustic modeling for indian languages: Iiit-h submission for low resource speech recognition challenge for indian languages, interspeech 2018," in *Proc. Interspeech 2018*, 2018, pp. 3192–3196. [Online]. Available: http://dx.doi.org/10.21437/Interspeech.2018-1584

[2] N. Fathima, T. Patel, M. C, and A. Iyengar, "Tdnn-based multilingual speech recognition system for low resource indian languages," 09 2018, pp. 3197–3201.

[3] B. Pulugundla, M. K. Baskar, S. Kesiraju, E. Egorova, M. Karafiát, L. Burget, and J. Černocký, "But system for low resource indian language asr," 09 2018, pp. 3182–3186.

[4] A. Biswas, E. Yılmaz, F. de Wet, E. van der Westhuizen, and T. Niesler, "Semi-supervised acoustic model training for five-lingual code-switched asr," 2019.

[5] A. Kannan, A. Datta, T. N. Sainath, E. Weinstein, B. Ramabhadran, Y. Wu, A. Bapna, Z. Chen, and S. Lee, "Large-scale multilingual speech recognition with a streaming end-to-end model," 2019.

[6] J. Chorowski, D. Bahdanau, K. Cho, and Y. Bengio, "End-to-end continuous speech recognition using attention-based recurrent nn: First results," 2014.

[7] W. Chan, N. Jaitly, Q. Le, and O. Vinyals, "Listen, attend and spell: A neural network for large vocabulary conversational speech recognition," in *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2016, pp. 4960–4964.

[8] T. Hori, S. Watanabe, Y. Zhang, and W. Chan, "Advances in joint ctc-attention based end-to-end speech recognition with a deep cnn encoder and rnn-lm," 2017.

[9] A. Graves, S. Fernández, F. Gomez, and J. Schmidhuber, "Connectionist temporal classification: Labelling unsegmented sequence data with recurrent neural 'networks," vol. 2006, 01 2006, pp. 369–376.

[10] A. Graves and N. Jaitly, "Towards end-to-end speech recognition with recurrent neural networks," in *Proceedings of the 31st International Conference on International Conference on Machine Learning - Volume 32*, ser. ICML'14. JMLR.org, 2014, p. II–1764–II–1772.

[11] J. Li, V. Lavrukhin, B. Ginsburg, R. Leary, O. Kuchaiev, J. M. Cohen, H. Nguyen, and R. T. Gadde, "Jasper: An end-to-end convolutional neural acoustic model," 2019.

[12] P. Golik, Z. Tüske, R. Schlüter, and H. Ney, "Convolutional neural networks for acoustic modeling of raw time signal in lvcsr," in *INTERSPEECH*, 2015.

[13] V. Passricha and R. K. Aggarwal, "Convolutional neural networks for raw speech recognition," in *From Natural to Artificial Intelligence*, R. Lopez-Ruiz, Ed. Rijeka: IntechOpen, 2018, ch. 2. [Online]. Available: https://doi.org/10.5772/intechopen.80026

[14] T. N. Sainath, R. J. Weiss, A. W. Senior, K. W. Wilson, and O. Vinyals, "Learning the speech front-end with raw waveform cldnns," in *INTERSPEECH*, 2015.

[15] D. Palaz, M. Magimai.-Doss, and R. Collobert, "Convolutional neural networks-based continuous speech recognition using raw speech signal," in *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2015, pp. 4295–4299.

[16] M. Ravanelli and Y. Bengio, "Speaker recognition from raw waveform with sincnet," in *2018 IEEE Spoken Language Technology Workshop (SLT)*, 2018, pp. 1021–1028.

[17] A. Graves, A. Mohamed, and G. Hinton, "Speech recognition with deep recurrent neural networks," in *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*, 2013, pp. 6645–6649.

[18] H. Sak, A. Senior, and F. Beaufays, "Long short-term memory recurrent neural network architectures for large scale acoustic modeling," *Proceedings of the Annual Conference of the International Speech Communication Association, INTERSPEECH*, pp. 338–342, 01 2014.

[19] S. Zhou, Y. Zhao, S. Xu, and B. Xu, "Multilingual recurrent neural networks with residual learning for low-resource speech recognition," in *INTERSPEECH*, 2017.

[20] A. Senior, H. Sak, and I. Shafran, "Context dependent phone models for lstm rnn acoustic modelling," in *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2015, pp. 4585–4589.

[21] S. A. Alim and N. K. A. Rashid, "Some commonly used speech feature extraction algorithms," in *From Natural to Artificial Intelligence*, R. Lopez-Ruiz, Ed. Rijeka: IntechOpen, 2018, ch. 1. [Online]. Available: https://doi.org/10.5772/intechopen.80419

[22] D. Povey, A. Ghoshal, G. Boulianne, L. Burget, O. Glembek, N. Goel, M. Hannemann, P. Motlicek, Y. Qian, P. Schwarz, J. Silovsky, G. Stemmer, and K. Vesely, "The kaldi speech recognition toolkit," in *IEEE 2011 Workshop on Automatic Speech Recognition and Understanding*. IEEE Signal Processing Society, Dec. 2011, iEEE Catalog No.: CFP11SRW-USB.

[23] B. Srivastava, S. Sitaram, R. Mehta, K. Mohan, P. Matani, S. Satpal, K. Bali, R. Srikanth, and N. Nayak, "Interspeech 2018 low resource automatic speech recognition challenge for indian languages," 08 2018, pp. 11–14.

[24] Wikipedia contributors, "Language — Wikipedia, the free encyclopedia," 2020, [Online; accessed 12-May-2020]. [Online]. Available: https://en.wikipedia.org/w/index.php?title=Language&oldid=955858850

[25] M. Ravanelli, T. Parcollet, and Y. Bengio, "The pytorch-kaldi speech recognition toolkit," in *In Proc. of ICASSP*, 2019.