REED: AN APPROACH TOWARDS QUICKLY BOOTSTRAPPING MULTILINGUAL ACOUSTIC MODELS

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

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

ABSTRACT

Multilingual automatic speech recognition (ASR) system is a single entity capable of transcribing multiple languages sharing a common phone space. Performance of such a system is highly dependent on the compatibility of the languages. State of the art speech recognition systems are built using sequential architectures based on recurrent neural networks (RNN) limiting the computational parallelization in training. This poses a significant challenge in terms of time taken to bootstrap and validate the compatibility of multiple languages for building a robust multilingual system. Complex architectural choices based on self-attention networks are made to improve the parallelization thereby reducing the training time. In this work, we propose Reed, a simple system based on 1D convolutions which uses very short context to improve the training time. To improve the performance of our system, we use raw time-domain speech signals directly as input. This enables the convolutional layers to learn feature representations rather than relying on handcrafted features such as MFCC. We report improvement on training and inference times by atleast a factor of $4\times$ and $7.4\times$ respectively with comparable WERs against standard RNN based baseline systems on SpeechOcean's multilingual low resource dataset.

Index Terms— multilingual automatic speech recognition, convolutional neural networks

1. INTRODUCTION

Multilingual automatic speech recognition (ASR) system is a single unit capable of transcribing speech utterances of multiple languages. Such systems have been extensively researched and have proven to be a viable solution for building robust speech recognition systems often outperforming the monolingual counterparts [1, 2]. One major application of a multilingual system is to build robust speech recognition systems for low resource languages where the system enables multiple languages to shoulder on each other and learn richer representations of their shared phone space [3–5]. Several techniques have been used to build such multilingual systems. Strategies like shared hidden layers [6], bottleneck features [7], multitask learning [8] train multiple monolingual acoustic models on multilingual data. One approach to building the monolingual models is to share the hidden representations between the languages while keeping the output layer language specific. Such systems are operated by either placing a Language Identification (LID) front-end to switch to the corresponding monolingual model or by selecting the best hypothesis obtained by running all the monolingual models. The performance of the former approach is largely dependent on the robustness and accuracy of the front-end LID system while the latter approach requires multiple monolingual acoustic models to be operated parallelly and thus is computationally expensive.

Major multilingual countries like India heavily engage in code switching [9]. It is a phenomenon of mixing multiple languages in a single utterance. Due to the language dependency in the aforementioned models, operating such models in a code-switched environment becomes very tricky. To employ large scale speech recognition systems in such multilingual countries, it is thus necessary to build language-agnostic systems. Joint multilingual speech recognition [3] is a technique where each of the component of the speech recognition system is language-agnostic. In such a system, a combined dataset is formed by combining the speech utterances, transcripts, corpus, and the phonetic space of each of the languages. The system is then trained in a monolingual fashion on the combined dataset.

The major challenge in building such a joint multilingual system is finding the right combination of compatible languages to work with on a given dataset [4]. This either requires an inherent understanding of different languages which creates a dependency on linguistic experts or calls for an iterative experimentation approach using different combinations of the languages. Today's state of the art speech recognition systems are built on recurrent neural networks [10–13]. The sequential nature of these systems limits the computational parallelization, thus increasing the computational time during training.

To improve the computational parallelization, complex architectures based on self-attention networks such as Transformers have been largely researched. [14] achieved state of the art results on LibriSpeech dataset using transformer based

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

acoustic model in a hybrid speech recognition system. [15] employed self-attention (SA) layer as a replacement for recurrent neural networks (RNN) in RNN-Transducer to build a SA-Transducer. However, transformers face several challenges, in practice it is found that the overall convergence time of the transformer is higher than the RNNs, transformers are also very hard to train as they are likely to get stuck in local optima [16]. Moreover, transformers often work well with very deep architectures [17] which in turn makes the architecture computationally expensive.

Time delayed neural networks (TDNN) have been widely accepted in the speech community to model long-term temporal dependencies while maintaining training time comparable to standard DNNs [18]. Standard acoustic models built using TDNN operate on long context windows to capture the long-term temporal dependencies. Phonemes, however, 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 [19]. Inspired by TDNN, we propose the use of 1D convolutional layers with very short context to capture short-term contextual dependencies. Using short-term context enables us to achieve significant boost in the training time. We use multi-channel outputs to capture richer short-term contextual phonetic representations.

Our work is motivated from [20] and [21]. We propose Reed, a hybrid fast multilingual speech recognition system. Our acoustic model is built on 1D convolutional layers operating on very short-term context. We improve the accuracy of the system by replacing the traditional handcrafted features such as MFCC with raw time-domain features as input to the convolutional layers. We do not perform any speaker normalization. This enables the model to learn features that are best suited for multilingual systems. To the best of our knowledge, our work is the first to use CNNs for building multilingual acoustic models operating directly on raw speech signals.

We use SpeechOcean's multilingual low resource dataset on three Indic languages, Gujarati, Tamil and Telugu to perform experiments with varying context window to balance the trade-off between the computational gain and the accuracy of the system. We compare its performance against a standard long short-term memory (LSTM) based baseline. To denote the effectiveness of raw-speech signals, we compare the performance of Reed against a model based on Reed trained on handcrafted MFCC features as a replacement for the raw speech input. Lastly, we apply Reed on different combination of the three languages to validate their compatibility. We use WER as the evaluation metric for all our experiments.

The rest of the paper is organized as follows. Section 2 describes convolutional neural networks and our proposed approach. Section 3 describes the experimental setup. Section 4 describes the experimental results with a comprehensive, comparative, and applicative analysis. Section 5 concludes our paper by stating the inferences drawn.

2. CONVOLUTIONAL NEURAL NETWORKS

In the speech community, speech recognition using convolutional neural networks have been widely researched. [22] used CNN to obtain 4-12% relative improvement over DNNs, on a 400-hr Broadcast News and 300-hr Switchboard task. [20] reported state of the art results on LibriSpeech dataset using very deep 1d convolutional layers with their deepest variant made up of 54 convolutions layers. [11] proposed novel CNN-RNN Transducer architecture featuring a fully convolutional encoder that incorporates global context information into convolution layers. [23] employed transformers and used convoltuional layers as a feature extraction layer to learn richer representation of the input acoustic feature. All the aforementioned approaches employed convolutional layers on handcrafted mel-filterbanks acoustic features.

CNNs have been widely adopted in the domain of computer vision as a feature extraction layer to learn meaningful representation of the raw image and video data which are often very high dimensional commonly in the range of 10^6. Training deep neural networks on such high dimensional data generates noise while tremendous computation power is needed to process such large fully connected layers. CNNs are used to reduce dimensionality of the raw data whilst retaining important spatial features such as contour boundaries, edges, simple curves, etc. Raw time-domain speech signals on the other hand, consist of 16000 features on just a 1-second long speech utterance with a frame rate of 16kHz. Each of these features represent a sample in time. However, limited research in the speech community has been done on employing convolutional layers directly on the raw speech signals.

[24] trains a fully convolutional network in an end-to-end fashion to predict characters from the raw waveform with an external language model to decode the words. [25] proposes a fully CNN based architecture called SincNet operating on raw speech signals that encourages the first convolutional layer to discover more meaningful filters by using parameterized sinc functions, which implement band-pass filters. In this work, we employ CNN as a feature extraction layer on raw speech signals and use a deep feed-forward network as the discriminative layer to build a fast multilingual acoustic model. Coupling this system with very short context as the input enables the system to achieve significant boost in the training time whilst maintaining comparable WERs against standard RNN based systems. The input to our system is raw time-domain speech signals with no speaker normalization.

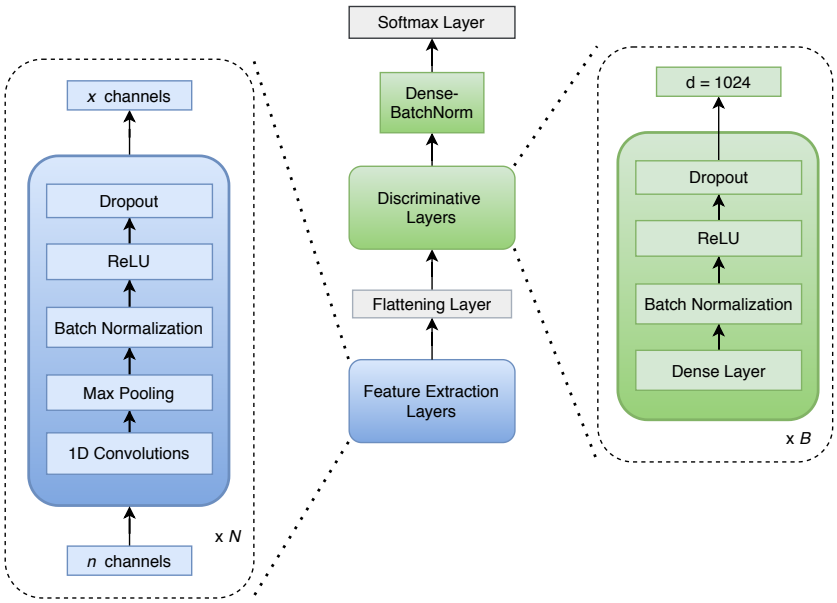

Fig. 1: *Reed: Proposed Acoustic Model; N - number of feature extraction layers, B - number of feed forward layers*

2.1. Context Window

Sequential models like RNNs capture coarticulation across consecutive phones. Such models are inherently complex and computationally expensive. 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 [19].

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 $(c_m + c_n + 1) \times f$ is passed, where c_m represents the number of context windows on the left, c_n is the number of context windows on the right and f is the frame length. In our work, we experiment with different context window sizes to find the optimal size for minimum loss in WER. The speech signals in our dataset have a frame rate of 16kHz and are sampled every 10 ms over a window of 25ms.

2.2. Proposed Model: Reed

Reed consists of two components: feature extraction layer using 1d convolutions and phone classification using deep feed forward network. Fig. 1 presents the design of our proposed model.

The feature extraction layer is composed of stacked 1d convolutional layers. Each layer performs a series of operations: convolution, max-pooling, batch-normalization, activation, and dropout. Each convolution operation is performed with a stride of 1. Each layer accepts a multi-channel input and outputs a multi-channel encoding. The first layer receives a single channel raw time-domain speech signal as input.

Mathematically, a non-strided convolution operation is expressed as:

$$C[a,b] = \sum_j \sum_l k[j,l] i[a+j, b+l] \quad (1)$$

where non-strided denotes that the stride for the filter k is 1. C is the output of the convolution operation, a and b are the convolution output indices, and i is multi-dimensional input. j and l are the dimensions of the filter.

The output dimension of the operation is given as:

$$d(C) = [p - j + 1, q - l + 1] \quad (2)$$

where p and q are the input dimensions.

The discriminative component is a stacked feed forward network where each layer performs the following: linear computation, batch-normalization, activation, and dropout. The output dimension of each layer is 1024 which is constant across all our experiments. The last layer is a linear layer with batch-normalization, no dropout and softmax activation to obtain the probability distribution across the phonetic space.

Reed accepts raw time-domain speech signals as input to the system along with the corresponding phonetic alignments and outputs the probability distribution across the phone space. In our experiments, we create the forced alignments using modified KneserNey smoothed tri-gram models and create the input time-domain features on a window of 25ms with a hop of 10ms. We then use a mixed language model

suitable for the purpose of code-switching to decode the phonetic probabilities into corresponding words. The unified language model is built by training a language model on the combined corpora of all the languages. We use tri-gram modified KneserNey language model trained using SRILM toolkit [26] included in Kaldi [27]. The acoustic model and the language model is trained in parallel and are combined in a hybrid fashion to make the system work end-to-end.

3. EXPERIMENTAL SETUP

3.1. Data

Our model is trained and computationally evaluated on the SpeechOcean's low-resource dataset [1] [28].

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

Languages	Train Set	Dev Set	Test Set
Gujarati	18307	4500	3075
Telugu	41682	3200	3040
Tamil	35231	3900	3081

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 [29]. 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 to overcome the challenge of data limitations. The data includes three Indic languages namely Gujarati, Tamil, and Telugu spoken by multiple speakers. The combined training data consists of 120 hours of spoken utterances with each language having approximately 40 hours of spoken utterances. The test and the validation data are 5 hours per language. Table 1 presents an overview of the number of utterances included in each language.

Text transcription along with the lexicon for the entire data is included in the dataset. We use a parser to convert utf8 text format to a language independent IT3 format [30]. The IT3 format text is then used to generate pronunciation sequences for all the words.

3.2. Toolkits

The Kaldi Speech recognition toolkit [27] has been used. SRILM toolkit is used for language modeling. The Wall Street Journal Kaldi Recipe is used for generating the alignments and for creating the MFCC features. We use 40-dimensional MFCC features without any speaker normalization. Kaldi LibriSpeech recipes are used to decode and score the system. Pytorch-Kaldi toolkit [31] was used for the development of all the mentioned acoustic models.

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

4. EXPERIMENTAL RESULTS

This section presents a comprehensive and comparative analysis between different architectural configurations of Reed against the baseline RNN to study the trade-off between the computational efficiency and WER degradation of the system. We compare Reed with a Reed based model trained on handcrafted MFCC features to denote of the effectiveness of the features learnt by the convolutional layers on the raw speech signals. Lastly, we present the applicative analysis by training Reed using different combination of the 3 low resource Indic languages to validate their compatibility.

4.1. Reed: Configurational Results

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 absolute WER degradation of different configurations against the baseline system (Exp 1) based on bidirectional LSTM. 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. The inference is drawn using a batch size of 1 to avoid parallelization across utterances. 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 architectural configurations in %, the numbers inside the braces indicate the number of left and right context windows, respectively.*

Models + Context	Gujarati	Telugu	Tamil
lstm + mfcc	**16.12**	**20.24**	**19.86**
cnn+raw + $\{0, 0\}$	24.06	31.23	30.94
cnn+raw + $\{-1, +1\}$	23.92	30.66	29.59
cnn+raw + $\{-2, +1\}$	20.13	26.80	25.73
cnn+raw + $\{-1, +2\}$	22.65	27.32	26.93
cnn+raw + $\{-2, +2\}$	19.48	22.37	21.55
cnn+raw + $\{-3, +2\}$	**18.36**	**21.24**	**20.92**
cnn+raw + $\{-2, +3\}$	19.02	22.33	20.98
cnn+mfcc + $\{-5, +5\}$	25.05	31.13	30.78

4.1.1. Baseline: Bidirectional LSTM

The baseline system is a hybrid architecture using the same language model and alignment technique as used in Reed. The acoustic model is a sequential model based on [32]. The

Table 3: *Performance comparison of average % WER, average absolute %WER degradation, training and inference times of different configurations against the baseline system (Exp 1) based on LSTM*

				Training		Inference	
Exp	Models + Context	Avg. WER	Avg. WER deg.	Time	Speed Up	Time	Speed Up
1	lstm + mfcc	**18.74**	-	$\sim$ 4.5 days	-	780 ms	-
2	cnn+raw + $\{0, 0\}$	28.74	-10	7.84 hours	$\sim 13.5\times$	15ms	$\sim 52\times$
3	cnn+raw + $\{-1, +1\}$	28.05	-9.31	11.56 hours	$\sim 9\times$	29ms	$\sim 26\times$
4	cnn+raw + $\{-2, +1\}$	24.22	-5.48	19.08 hours	$\sim 5.6\times$	30ms	$\sim 26\times$
5	cnn+raw + $\{-1, +2\}$	25.63	-6.89	20.63 hours	$\sim 5\times$	32ms	$\sim 24\times$
6	cnn+raw + $\{-2, +2\}$	21.13	-2.39	24.50 hours	$\sim 4.4\times$	89ms	$\sim 8.7\times$
7	cnn+raw + $\{-3, +2\}$	**20.17**	-1.43	27.29 hours	$\sim 4\times$	105ms	$\sim 7.4\times$
8	cnn+raw + $\{-2, +3\}$	20.77	-2.03	28.62 hours	$\sim 3.5\times$	109ms	$\sim 7.15\times$
9	cnn+mfcc + $\{-5, +5\}$	28.98	-10.24	3.55 hours	$\sim 30\times$	12 ms	$\sim 65\times$

model is trained on 40 dimensional MFCC features without any speaker normalization to make the architecture comparable to Reed. The LSTM has 4 hidden layers with 600 hidden units in each layer. In a bi-directional setting, the total hidden units in each layer will be $2\times600 = 1200$. A batch size of 8 is used to keep the GPU memory from overflowing. A learning rate of 5×10^{-4} is used during training. A dropout of 50% is employed in each layer. ReLU activation is used in each layer except the final layer which uses a softmax activation to output the probability distribution. The system is trained using Negative Log Likelihood (NLL) loss function with Adam optimizer.

4.1.2. Reed trained on Raw Speech Signals

The feature extraction layer is composed of 3 hidden layers. The 1st, 2nd and 3rd layer has 8, 64 and 128 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 8×10^{-4} is employed with a halving factor of 0.5 and an improvement threshold of 0.001. ReLU as an activation is employed at each layer.

The deep discriminative feed-forward network consists of 5 hidden layers. Each of the hidden layers except the output layer is made up of 1024 units with a dropout of 10% and ReLU activation. Each of the layers in the acoustic model employ batch normalization. A learning rate of 4×10^{-4} is applied with a halving factor of 0.5 and an improvement threshold of 0.001. The final layer employs no dropout with a softmax activation to obtain the probability distribution. The system is trained using NLL loss function with RMSprop optimizer. Unlike the LSTM based baseline model, the GPU is able to fit mini-batches of size 64 during training improving the computational efficiency.

The speech utterances used in the experiments have a sampling rate of 16kHz. Sampling frames on a window of 25ms fetches a 400 dimensional feature vector. Varying the context window from no-context to 3 contexts on either side increases the feature dimension upto a total of $(3 + 3 + 1) \times 4 = 2800$ dimension. Our best model achieves a WER of 18.36, 21.24, and 20.92 on Gujarati, Telugu, and Tamil respectively with 3 left and 2 right context window (Exp 7, Table 3). The results are comparable to the baseline with an average relative degradation of only $\sim 7\%$. A computation boost of $4\times$ in training time and $7.4\times$ in inference time is obtained with this configuration taking only an average of 105ms to infer on an average of 5.85s long speech utterance compared to 780ms on the baseline (Exp 1).

Although an obvious observation, Table 3 presents an interesting trend of the decrease in training and inference time with the decrease in the context window. There is a significant jump in the WER between Exp 4 and Exp 3 suggesting no-context or a context window of 1 on either side doesn't capture the relevant coarticulations. It can also be seen that a greater context on the left produces better WER compared to greater context on the right for the same context window size with comparable training and inference time speed up (Exp 7-Exp 8; Exp 4-Exp 5) indicating that the left context contains more relevant information compared to the right context. Depending on the requirement of the system and accepted WER degradation, Reed with context window varying from $\{-2, +1\}$ to $\{-3, +2\}$ (Exp 5 - Exp 7) would be the optimal choices for quickly bootstrapping a system.

4.1.3. Reed trained on MFCC features

Exp 9 in Table 3 employs the same configuration mentioned in Section 4.1.2. The two differences in the system are: Only the first layer of the feature extraction layers is used, hand-

crafted 40-dimensional MFCC features are used as a replacement for the raw speech inputs. No speaker normalization is used to keep the architecture comparable to Reed. The features are computed on a window size of 25ms with a hop of 10ms. A standard context window of $\{-5, +5\}$ is used to train and draw inference on the system. This configuration achieves the highest boost of $30\times$ and $65\times$ in the training and inference time respectively. However, this comes with a significant degradation in the WER with a relative degradation of $\sim 50\%$. The table presents an interesting observation that even though the MFCC based system receives a total context window of $5 + 5 = 10$, its performance in terms of WER degradation is at par with the most basic Reed trained on no-context (Exp 2). This validates our hypothesis that CNNs can capture richer representation of the raw speech signals than traditional handcrafted MFCC features.

4.2. Applicative Linguistic Results

In this section, we present the analysis drawn on different combinations of the three Indic languages. We employ Reed with a context window of $\{-3, +2\}$ to build all the models mentioned in this section.

The three Indic languages can be categorized into two family of languages: Indo-Aryan and Indo-Dravidian. Gujarati belongs to Indo-Aryan while Tamil and Telugu belong to Indo-Dravidian. We believe that the performance of a multilingual system is directly dependent on the compatibility of the languages. We hypothesize that using a combination of languages belonging to the same language family should give us better WERs compared to a system built by combining the languages belonging to different language families.

Table 4: *%WER of different combination of languages; gu-Gujarati, te-Telugu, ta-Tamil*

		Multilingual	
Languages	Monolingual	$ta + te$	$ta + te + gu$
Gujarati	19.88	-	**18.36**
Telugu	29.07	**20.85**	21.24
Tamil	28.91	**20.67**	20.92

To validate our hypothesis, we start by building a monolingual system for each of the three languages and note down the monolingual WERs. We then combine the two Dravidian languages - Tamil and Telugu (Model $ta + te$, Table 4). By doing so, we increase the phonemes overlap across these languages. As observed in Table 4, the performance of the two languages improve compared to their monolingual counterparts. In our next experiment, we add Gujarati, which belongs to Indo-Aryan family, to the combined dataset of Tamil amd Telugu (Model $ta + te + gu$). We observed that even though the WER of all three languages decreased as compared to their monolingual counterparts, the WERs of Tamil and Telugu increased as compared to Model $ta + te$. This suggests the possibility that familial correspondence maybe more important than additional data provided by the pooled languages. Similar results were also reported in [4].

This validates our hypothesis that even though combining languages with shared phone space can improve the accuracy of the individual languages, smart selection of languages used to build the multilingual system can further decrease the WERs for individual languages.

These combinations were validated in a short span of three days on a single GPU.

5. CONCLUSION

In this work, we propose a fast-multilingual system, Reed, for quickly bootstrapping and validating the compatibility of different languages for building a robust multilingual system. Reed is a hybrid system based on a simple acoustic model which uses only 1D convolutional layers and feed-forward networks and operates on very short context. Experimental results demonstrate that CNNs as a feature extraction layer can be used to learn rich representation of the raw speech signals instead of relying on traditional handcrafted MFCC features. Our most optimal model provides a training and inference boost of $4\times$ and $7.4\times$ respectively with a relative WER degradation of only $\sim 7\%$ against the baseline. We also present a comparative study between the variation in the context window size, the boost in training time and the average WER degradation against the baseline. Depending on the requirement of the system and acceptable WER degradation, the study can be used to employ Reed to build robust multilingual systems without additional linguistic knowledge.

6. ACKNOWLEDGEMENTS

We would like to thank TDIL MeitY for supporting this work through "Crowd Sourcing of Large Speech Data Sets To Enable Indian Language Speech - Speech Solutions (Pilot Project)". We would also like to thank Rajeev Gupta, Sandipan Dandapat, and Sunayana Sitaram who are researchers at Microsoft, for their valuable feedback.

7. REFERENCES

[1] Astik Biswas, Emre Yılmaz, Febe de Wet, Ewald van der Westhuizen, and Thomas Niesler, "Semi-supervised acoustic model training for five-lingual code-switched asr," 2019.

[2] Anjuli Kannan, Arindrima Datta, Tara N. Sainath, Eugene Weinstein, Bhuvana Ramabhadran, Yonghui Wu,

Ankur Bapna, Zhifeng Chen, and Seungji Lee, "Large-scale multilingual speech recognition with a streaming end-to-end model," 2019.

[3] Hari Krishna, Krishna Gurugubelli, Vishnu Vidyadhara Raju V, and Anil Kumar 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.

[4] Noor Fathima, Tanvina Patel, Mahima C, and Anuroop Iyengar, "Tdnn-based multilingual speech recognition system for low resource indian languages," 09 2018, pp. 3197–3201.

[5] Bhargav Pulugundla, Murali Karthick Baskar, Santosh Kesiraju, Ekaterina Egorova, Martin Karafiát, Lukas Burget, and Jan Černocký, "But system for low resource indian language asr," 09 2018, pp. 3182–3186.

[6] A. Ghoshal, P. Swietojanski, and S. Renals, "Multilingual training of deep neural networks," in *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*, 2013, pp. 7319–7323.

[7] Samuel Thomas, Sriram Ganapathy, and Hynek Hermansky, "Multilingual mlp features for low-resource lvcsr systems," 03 2012, pp. 4269–4272.

[8] D. Chen and B. K. Mak, "Multitask learning of deep neural networks for low-resource speech recognition," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 23, no. 7, pp. 1172–1183, 2015.

[9] Emre Yılmaz, Henk van den Heuvel, and David Van Leeuwen, "Code-switching detection using multilingual dnns," 12 2016.

[10] William Chan and Ian Lane, "Deep recurrent neural networks for acoustic modelling," 2015.

[11] Wei Han, Zhengdong Zhang, Yu Zhang, Jiahui Yu, Chung-Cheng Chiu, James Qin, Anmol Gulati, Ruoming Pang, and Yonghui Wu, "Contextnet: Improving convolutional neural networks for automatic speech recognition with global context," 2020.

[12] Jan Chorowski, Dzmitry Bahdanau, Dmitriy Serdyuk, Kyunghyun Cho, and Yoshua Bengio, "Attention-based models for speech recognition," 2015.

[13] Awni Hannun, Carl Case, Jared Casper, Bryan Catanzaro, Greg Diamos, Erich Elsen, Ryan Prenger, Sanjeev Satheesh, Shubho Sengupta, Adam Coates, and Andrew Y. Ng, "Deep speech: Scaling up end-to-end speech recognition," 2014.

[14] Yongqiang Wang, Abdelrahman Mohamed, Due Le, Chunxi Liu, Alex Xiao, Jay Mahadeokar, Hongzhao Huang, Andros Tjandra, Xiaohui Zhang, Frank Zhang, and et al., "Transformer-based acoustic modeling for hybrid speech recognition," *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, May 2020.

[15] Zhengkun Tian, Jiangyan Yi, Jianhua Tao, Ye Bai, and Zhengqi Wen, "Self-attention transducers for end-to-end speech recognition," *Interspeech 2019*, Sep 2019.

[16] Shigeki Karita, Nelson Enrique Yalta Soplin, Shinji Watanabe, Marc Delcroix, Atsunori Ogawa, and Tomohiro Nakatani, "Improving Transformer-Based End-to-End Speech Recognition with Connectionist Temporal Classification and Language Model Integration," in *Proc. Interspeech 2019*, 2019, pp. 1408–1412.

[17] Ngoc-Quan Pham, Thai-Son Nguyen, Jan Niehues, Markus Müller, Sebastian Stüker, and Alexander Waibel, "Very deep self-attention networks for end-to-end speech recognition," 2019.

[18] Vijayaditya Peddinti, D. Povey, and S. Khudanpur, "A time delay neural network architecture for efficient modeling of long temporal contexts," in *INTERSPEECH*, 2015.

[19] 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.

[20] Jason Li, Vitaly Lavrukhin, Boris Ginsburg, Ryan Leary, Oleksii Kuchaiev, Jonathan M. Cohen, Huyen Nguyen, and Ravi Teja Gadde, "Jasper: An end-to-end convolutional neural acoustic model," 2019.

[21] Pavel Golik, Zoltán Tüske, Ralf Schlüter, and Hermann Ney, "Convolutional neural networks for acoustic modeling of raw time signal in lvcsr," in *INTERSPEECH*, 2015.

[22] T. N. Sainath, A. Mohamed, B. Kingsbury, and B. Ramabhadran, "Deep convolutional neural networks for lvcsr," in *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*, 2013, pp. 8614–8618.

[23] L. Dong, S. Xu, and B. Xu, "Speech-transformer: A no-recurrence sequence-to-sequence model for speech recognition," in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2018, pp. 5884–5888.

[24] Neil Zeghidour, Qiantong Xu, Vitaliy Liptchinsky, Nicolas Usunier, Gabriel Synnaeve, and Ronan Collobert, "Fully convolutional speech recognition," 2018.

[25] Mirco Ravanelli and Yoshua Bengio, "Speaker recognition from raw waveform with sincnet," 2018.

[26] Andreas Stolcke, "Srilm — an extensible language modeling toolkit," *Proceedings of the 7th International Conference on Spoken Language Processing (ICSLP 2002)*, vol. 2, 07 2004.

[27] Daniel Povey, Arnab Ghoshal, Gilles Boulianne, Lukas Burget, Ondrej Glembek, Nagendra Goel, Mirko Hannemann, Petr Motlicek, Yanmin Qian, Petr Schwarz, Jan Silovsky, Georg Stemmer, and Karel Vesely, "The kaldi speech recognition toolkit," in *IEEE 2011 Workshop on Automatic Speech Recognition and Understanding*. Dec. 2011, IEEE Signal Processing Society, IEEE Catalog No.: CFP11SRW-USB.

[28] Brij Srivastava, Sunayana Sitaram, Rupesh Mehta, Krishna Mohan, Pallavi Matani, Sandeepkumar Satpal, Kalika Bali, Radhakrishnan Srikanth, and Niranjan Nayak, "Interspeech 2018 low resource automatic speech recognition challenge for indian languages," 08 2018, pp. 11–14.

[29] Wikipedia contributors, "Language — Wikipedia, the free encyclopedia," 2020, [Online; accessed 12-May-2020].

[30] Arun Baby, Nishanthi N. L., Anju Leela Thomas, and Hema A. Murthy, "A unified parser for developing indian language text to speech synthesizers," in *Text, Speech, and Dialogue - 19th International Conference, TSD 2016, Brno, Czech Republic, September 12-16, 2016, Proceedings*, Petr Sojka, Ales Horák, Ivan Kopecek, and Karel Pala, Eds. 2016, vol. 9924 of *Lecture Notes in Computer Science*, pp. 514–521, Springer.

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

[32] Duc Le, Xiaohui Zhang, Weiyi Zheng, Christian Fügen, Geoffrey Zweig, and Michael L. Seltzer, "From senones to chenones: Tied context-dependent graphemes for hybrid speech recognition," 2019.