Interspeech 2018
2-6 September 2018, Hyderabad

ISI ASR System for the Low Resource Speech Recognition Challenge for Indian Languages

Jayadev Billa

Information Sciences Institute,
University of Southern California, Marina del Rey, CA 90292, USA
jbilla@isi.edu

Abstract

This paper describes the ISI ASR system used to generate ISI's submissions across Gujarati, Tamil and Telugu speech recognition tasks as part of the Low Resource Speech Recognition Challenge for Indian Languages. The key constraints on this task were limited training data, and the restriction that no external data be used. The ISI ASR system leverages our earlier work on data augmentation and dropout approaches and current work on multilingual training within a Eesen based end-to-end Long Short Term Memory (LSTM) based automatic speech recognition (ASR) system trained with the Connectionist Temporal Classification (CTC) loss criterion, and demonstrates, to the best of our knowledge, one of the first times such systems have been applied to low resource languages with performance comparable and some cases better than hybrid DNN systems.

Our best monolingual systems show between 6.5% to 25.5% relative reduction in word error rate (WER) compared to the challenge organizer's Time Delay Neural Network (TDNN) based baseline WERs. We further extend these systems with multilingual training approaches that lead to an additional 4.5% to 11.1% relative reduction in WER as measured on the development set.

Index Terms: speech recognition, LSTM, CTC, low-resource ASR, multilingual learning

1. Introduction

The Low Resource Speech Recognition Challenge for Indian Language consists of a speech recognition task in three Indian languages, Gujarati, Tamil and Telugu, with 40 hours of transcribed audio data per language and a development set consisting of 5 hours of transcribed audio per language[1]. The key constraint in this challenge, apart from the low amounts of training data, was the exclusion of external data beyond what is provided as part of the challenge. For each language, lexicons with phonetic pronunciations were provided in two phoneme sets, the CMU Indic Frontend [1] as well as the IITM Common Label (IITM-C) set [2], with the choice of which lexicon to use left to the participants.

We have been investigating speech recognition approaches for low resource languages as part of our ongoing efforts in the IARPA Machine Translation for English Retrieval of Information in Any Language (MATERIAL) program [3]; this challenge offered an opportunity to extend our early work to Indian languages and explore the robustness of our modeling approach. Note that apart from a rudimentary familiarity with Telugu, the author has no expertise in the other languages, Gujarati and Tamil; as such, none of the work presented here required or used any language specific knowledge or expertise.

[1] Data provided by SpeechOcean.com and Microsoft.

In recent years there has been quite a bit of interest in speech recognition in low resource languages [4]. The key approaches to address the general paucity of target language data include data augmentation [5, 6, 7, 8, 9], semi-supervised training, e.g., [10, 11, 12], Multilayer Perceptron (MLP) based feature extraction [6, 12, 13, 14], cross-lingual knowledge transfer [15, 16], multi-task [17, 18], multilingual training [19, 20, 21] and multilingual phoneme sets [22]. Data augmentation seeks to increase the amount of available training data by transforming the original data to generate additional data; examples of data transformations include speed perturbation [9], vocal tract length perturbation [5], and speaker transformation [8]. Semi-supervised training is similar to data augmentation in that in seeks to increase available training data, but does so by using a bootstrap ASR model to transcribe any available untranscribed data, and adding the resulting transcripts, with the highest confidence in accuracy, to the training set e.g., [10, 11, 12]. MLP based feature extraction involves using a neural network trained on larger datasets to generate features, referred to as bottleneck features, with the implicit assumption that the initial layers of a neural network extract a language agnostic representation of speech features that would be helpful under low resource conditions. The validity of this general approach is supported by results across multiple teams [6, 12, 13, 14]. Multi-task training, on the other hand, attempts to increase the effectiveness of the available data by training on different but related tasks at the same time, in effect, guiding the model to model the underlying latent features common to the target tasks, see [18] for low resource language efforts and [17] for a general treatment. Another class of approaches addresses the lack of data by combining available data across languages via a seed model for the low resource language, referred to as cross-lingual transfer [15, 16], or directly training a multilingual system on multiple languages which include the target language [19, 20, 21]. A similar related approach is to train on a set of languages with a common multilingual phoneme set and use the trained model on an entirely different language, e.g. [22]

It should be noted that the majority of these approaches for low resource languages were implemented in the general framework of a hybrid DNN ASR system, where the neural network component, trained with cross-entropy loss, generates posterior probabilities that are then combined, in most cases, with an hidden markov model to capture the time element of speech, to generate output speech recognition text. Our efforts, on the other hand, use a recurrent neural network model, in particular, a long short-term memory (LSTM) based neural network model, that jointly models the features and their temporal context, trained with the connectionist temporal classification (CTC) loss function [23]. CTC directly estimates the probability of the output sequence given the input sequence of speech features. Since CTC is a sequence to sequence loss measure

10.21437/Interspeech.2018-2473

there is no need for frame level alignments for training, unlike the cross-entropy loss which requires frame level alignments. An LSTM-CTC ASR system can therefore be trained as an end-to-end sequence-to-sequence system using backpropagation with no additional data beyond the input speech features and the output phoneme, grapheme, or word sequence.

Initial efforts to use recurrent neural network based end-to-end ASR systems were restricted to domains with an abundance of training data, measured in the thousands of hours, where these systems outperformed comparable hybrid DNN systems [24, 25]. However, until recently these systems lagged comparable hybrid DNN systems when trained on smaller training sets (e.g., discussion in [26]). In [27, 28], we demonstrated that with suitable data augmentation, feature presentation coupled with dropout on recurrent and feedforward connections, LSTM-CTC ASR systems could be competitive with hybrid DNN systems with 100-250 hours of data. In this paper, we demonstrate that LSTM-CTC ASR systems can be competitive even with 40-50 hours of data. To the best of our knowledge, this is the first time that LSTM-CTC ASR systems have been built entirely on low resource languages without access to additional data, though in [29], LSTM-CTC systems trained on larger data sets are adapted to low resource languages.

In this paper, we start in Section 2 with a description of our canonical LSTM-CTC ASR system architecture and detail design decisions. Section 3 presents the results of applying this system architecture in the context of monolingual training. Section 4 details our efforts to overcome data paucity using multilingual training, with Section 5 covering additional attempts to further improve on the multilingual systems with fine tuning and retraining on the target language data. Section 6 provides a brief summary of our submission and conclusion.

2. LSTM-CTC ASR System Architecture

The LSTM-CTC ASR system we have developed is based the publicly available Eesen Toolkit [30]. The acoustic model in Eesen is a deep bidirectional LSTM neural network, trained with the CTC loss function, which minimizes the negative log summed probability of the correct label sequence given the input sequence, via a forward-backward algorithm that sums across all possible alignments. Details on the CTC loss function can be found in [23]. Eesen uses a weighted finite state grammar (WFST) to incorporate the language model and generate the final word sequence from the network outputs. Details can be found in [30]. The vector formulas that describe the LSTM cell are

$$\mathbf{i}_t = \sigma(\mathbf{W}_i\mathbf{x}_t + \mathbf{R}_i\mathbf{h}_{t-1} + \mathbf{P}_i\mathbf{c}_{t-1} + \mathbf{b}_i) \quad (1)$$

$$\mathbf{f}_t = \sigma(\mathbf{W}_f\mathbf{x}_t + \mathbf{R}_f\mathbf{h}_{t-1} + \mathbf{P}_f\mathbf{c}_{t-1} + \mathbf{b}_f) \quad (2)$$

$$\mathbf{c}_t = \mathbf{f}_t \odot \mathbf{c}_{t-1} + \mathbf{i}_t \odot \phi(\mathbf{W}_c\mathbf{x}_t + \mathbf{R}_c\mathbf{h}_{t-1} + \mathbf{b}_c) \quad (3)$$

$$\mathbf{o}_t = \sigma(\mathbf{W}_o\mathbf{x}_t + \mathbf{R}_o\mathbf{h}_{t-1} + \mathbf{P}_o\mathbf{c}_t + \mathbf{b}_o) \quad (4)$$

$$\mathbf{h}_t = \mathbf{o}_t \odot \phi(\mathbf{c}_t) \quad (5)$$

where $\mathbf{x}_t$, $\mathbf{o}_t$ and $\mathbf{h}_t$ respectively represent the input, cell memory and cell output vectors at time t, $\mathbf{W}$ are rectangular input weight matrices connecting inputs to the LSTM cell, $\mathbf{R}$ are square recurrent weight matrices connecting the previous memory cell state to the LSTM cell, $\mathbf{P}$ are diagonal peephole weight matrices and $\mathbf{b}$ are bias vectors. Functions σ and ϕ are the *logistic sigmoid* and $\tanh$ nonlinearities respectively. Operator $\odot$ represents the point-wise multiplication of two vectors.

These cells are then arranged into a bidirectional layer where data is processed independently in the forward and backward directions [31]. The outputs from both forward and backward directions are concatenated to form the input to the next recurrent layer.

$$\mathbf{y}_t = [\overrightarrow{\mathbf{h}}_t, \overleftarrow{\mathbf{h}}_t] \quad (6)$$

The LSTM model in our experiments consists of 4 bidirectional stacked layers of 640 LSTM cells (320 in each direction). The base input features consist of 40 dimensional mel warped filterbank features with Δ and $\Delta\Delta$ features using a 25ms window and 10ms frame rate, normalized with per utterance means subtraction and variance normalization[2]. The model outputs correspond to the context independent phonemes. The base features are further stacked and strided to create composite frames, consisting of three base frames, with no frame overlap, across consecutive composite frames, resulting in a nominal 30ms frame rate for the system as a whole. The salient elements of our baseline system are our data augmentation strategy and dropout approach. Both of these approaches are covered in detail in [27, 28] but we provide a high level overview below for completeness.

2.1. Data Augmentation

Our approach to data augmentation, max perturbation [27], essentially creates a 9 fold expansion of the available data by permutating vocal tract length normalization (VTLN) and feature frame rate factors during base feature generation. In particular, we use VTLN factors in $[0.8, 1.0, 1.2]$ and vary the feature frame rate in $[8\text{ms}, 10\text{ms}, 11\text{ms}]$, creating a total of 9 variants of the data. In early work [27], we found that max perturbation significantly outperforms speed perturbation; since then we have explored both approaches across a variety of languages and corpora and discovered that, by tuning speed rates, speed perturbation can largely match max perturbation performance. We continue to use max perturbation since it almost always provides the best performance without additional tuning.

2.2. LSTM Dropout

The implementation of dropout within the ISI ASR system mirrors the sequence-level, stochastic dropout approach detailed in [27, 28]. At a high level, we implement dropout on the feedforward connections i.e., the output of each LSTM layer, as well as within each LSTM cell, on the LSTM cell update i.e., recurrent dropout without memory loss following [32]. In the case of feedforward dropout, Equation 6 becomes

$$\mathbf{y}_t = \mathbf{m}_f \odot [\overrightarrow{\mathbf{h}}_t, \overleftarrow{\mathbf{h}}_t] \quad (7)$$

where $\mathbf{m}_f$ is the dropout mask applied to the output of the LSTM layer. For recurrent dropout, Equation 3 changes to

$$\begin{aligned}\mathbf{c}_t =& \mathbf{f}_t \odot \mathbf{c}_{t-1} \\ &+ \mathbf{m}_r \odot \mathbf{i}_t \odot \phi(\mathbf{W}_c\mathbf{x}_t + \mathbf{R}_c\mathbf{h}_{t-1} + \mathbf{b}_c)\end{aligned} \quad (8)$$

where $\mathbf{m}_r$ is the dropout mask applied to the cell update within the LSTM cell. In both dropout paradigms, we keep the dropout mask fixed across all timesteps in an utterance, and vary the mask on a per utterance basis. During training, for each minibatch, we determine whether to apply feedforward or recurrent dropout based on an equiprobable Bernoulli distribution, i.e., an unbiased coin toss. The dropout rate is set to 0.2, i.e., 20%

[2] In general, we apply per speaker means subtraction and variance normalization, but in this challenge speaker detail was not available.

Table 1: *Phoneme counts across phoneme sets.*

	Phoneme Set		
Language	CMU-Indic	IITM-C	IITM-CR
Gujarati	54	50	32
Tamil	38	37	30
Telugu	56	49	33
Combined	60	57	38

of connections are masked during training, across all our experiments – we did not attempt to optimize this meta parameter. See [27, 28] for a more detailed exposition of our dropout paradigm and related experiments.

2.3. Phoneme Set

One early choice in this challenge was which phoneme set to base our systems on. In earlier work [27], we observed a small improvement in performance with a reduced phoneme set, where we collapsed phoneme stress variants, on Librispeech with LSTM-CTC models. These earlier experiments biased our preference towards more compact phoneme sets. Initial review of the phoneme sets indicated that the IITM-C set was slightly more compact than the CMU-Indic set. Table 1 summarizes the phoneme counts across both phoneme sets. Motivated by the same experience, we also wanted to explore if an even more compact representation would be beneficial to system performance. To this end, we reduced the IITM-C set by removing phoneme suffixes [2]: *h* indicating aspiration, *x* indicating retroflex place of articulation, *q* corresponding to a nukta/bindu, and *n* indicating vowel nasalization, in the lexicon. This reduced set, IITM-CR in Table 1, reduced the combined phoneme set by 33% to 38 phonemes. One drawback of phoneme reduction, however, as applied to the IITM-C set, is that the native script is no longer readily recoverable from the transliteration.

3. Monolingual Training

To establish an LSTM-CTC baseline for each language, we trained separate systems for each language using the IITM-C and IITM-CR phoneme sets. The performance of these systems on the development and evaluation sets are summarized in Table 2. In all experiments, we use a trigram WFST language model, generated from the corresponding language training transcripts. During the course of the challenge, the organizers updated the lexicons for all languages; given limited time, we opted to continue training all systems with the original lexicon, instead of restarting training, and decode using the updated lexicon. For completeness, after the challenge was concluded, we reran the baseline system using the CMU-Indic phoneme set, as well as scored all systems on the evaluation test set. These results are also included in Table 2. For comparison, the challenge organizer's CMU-Indic based baseline results on the development set with Kaldi TDNN systems are shown in Table 3. Our monolingual systems for Gujarati and Telugu outperform the challenge baselines, with significant over performance in Gujarati on the order of ≈25% relative reduction in WER. The identically trained Tamil system, however, lagged the challenge baseline with a 4-5% relative increase in WER. Following the challenge and initial submission of this paper, we discovered that using the updated lexicon for Tamil training resulted

Table 2: *Monolingual LSTM-CTC system results (%WER).*

		Phoneme Set		
Test	Language	CMU-Indic	IITM-C	IITM-CR
Dev	Gujarati	15.26	14.73	14.98
	Tamil	18.18	20.51	20.25
	Tamil†		19.27	19.08
	Telugu	20.10	20.03	19.87
Eval	Gujarati	21.80	20.91	21.44
	Tamil	18.00	20.31	20.44
	Tamil†		18.62	18.77
	Telugu	20.04	19.77	19.60

† Trained with updated lexicon.

Table 3: *Development set baselines from organizers (%WER).*

Language	CMU-Indic
Gujarati	19.76
Tamil	19.45
Telugu	22.61

in performance on par with the challenge baseline, also in Table 2. Across all three languages we observe that the IITM-C and IITM-CR based systems are largely equivalent in performance – suggesting that a simplified phoneme set would suffice for these languages in particular and perhaps other Indian languages as well.

The CMU-Indic phoneme set performs considerably better on Tamil but lags on Gujarati and Telugu in comparison with the IITM-C phoneme sets, and explains, in part, why our Tamil system performance did not show similar WER improvement over the organizer's baseline as the Gujarati and Telugu systems. That said, comparing the challenge organizer's TDNN model vis-à-vis the CMU-Indic based LSTM-CTC Tamil system, we note that the LSTM-CTC system demonstrates a 6.53% relative reduction in WER on the development set.

4. Multilingual Training

In this challenge, the restriction of no external data precluded the option of cross-lingual language transfer, where one uses a model, trained in a language with more data than is available for the low resource language, as an initial seed model to train the target language system e.g., [13, 21]. Alternatively, one can train a multilingual system by combining the data from multiple languages. This challenge is clearly more conducive toward the latter, multilingual training approach.

A number of approaches to multilingual training have been proposed e.g., [16, 19, 20, 21]. The typical approach involves a DNN model sharing the input and all hidden layers with a separate output layer for each language e.g., [16, 20, 21]. An alternate simpler approach is a model in which all layers, including the output layer, are shared. In this model, the output units cover the superset of phonemes in the component languages. This latter approach has the additional advantage of increasing training data for phonemes that overlap across languages, which would be beneficial given the overall scarcity of data within and across languages. In light of these benefits, we opted for fully shared layers for multilingual modeling in our experiments.

Our implementation is fairly straightforward: we pool data

Table 4: *Multilingual LSTM-CTC system results (%WER).*

Test	Language	Phoneme Set	
		IITM-C	IITM-CR
Dev	Gujarati	13.32	13.24
	Tamil	19.45	20.87
	Telugu	18.86	18.54
Eval	Gujarati	19.33	19.30
	Tamil	19.61	20.92
	Telugu	18.56	18.86

across the three languages, Gujarati, Tamil and Telugu; the model output corresponds to the superset of phonemes in these languages. Since the available training data were similar in size, we did not attempt to explore language weighting by changing the relative amount of training per language within a training epoch [19]. During decoding, we apply the language specific WFST language model (LM) as as opposed to a pooled LM since the test language is known.

Table 4 summarizes the results from training on the pooled data set, with the original lexicon, and decoding with the language specific WFST grammar. Across both development and evaluation sets, except for the Tamil IITM-CR model, we see a consistent improvement with the pooled models over the monolingual baseline systems. One can also ask if there are better approaches to pooling, e.g. should only familial languages be pooled? From the results, while Telugu and Tamil belong to the same Dravidian family and Gujarati does not, Gujarati shows a sizable WER reduction: 9.57% on development, 7.56% on evaluation. This indicates that with small training data sets, familial correspondence is less important than the additional training data provided by the pooled languages.

5. Fine Tuning and Retraining

An interesting consequence of the multilingual training described in Section 4 is that we now have a model trained on more data than each target language, a prerequisite for cross-lingual knowledge transfer. Strictly speaking, the cross-lingual knowledge transfer paradigm would be more exact if we train a multilingual system on two languages and use that as a seed model for the third language.

Given time constraints we decided to proceed with the three language multilingual model as a seed model with two training approaches. In one approach, fine tuning, we use the unaugmented data in each language to train the corresponding monolingual model starting from the multilingual model. Alternatively, we can train with the full augmented data set and retrain a monolingual model from the multilingual seed model.

Fine tuning and full retraining results, using the original lexicon during training, are summarized in Table 5 and Table 6 respectively. Across both approaches, we see only minor changes in WER in all three languages. This indicates that at this operating point, a multilingual model captures all the available language specific discriminative detail and further retraining provides no additional gain. It would be interesting to see if we can improve on the pooled multilingual system with a more classical cross-lingual knowledge transfer paradigm by training on two languages to create a multilingual model which is then trained on the third language. Given the small training data size, following the results in Section 4, the choice of which languages to pool, familial or otherwise, may not be as crucial as it might be with a larger training data set.

Table 5: *Fine tuning multilingual model results (%WER).*

Test	Language	Phoneme Set	
		IITM-C	IITM-CR
Dev	Gujarati	13.10	13.28
	Tamil	19.59	20.84
	Telugu	18.90	18.63
Eval	Gujarati	19.22	19.73
	Tamil	19.72	20.92
	Telugu	18.74	18.95

Table 6: *Retraining with multilingual model as seed (%WER).*

Test	Language	Phoneme Set	
		IITM-C	IITM-CR
Dev	Gujarati	13.54	13.24
	Tamil	19.59	20.94
	Telugu	18.94	18.61
Eval	Gujarati	19.23	19.11
	Tamil	19.80	20.82
	Telugu	18.77	18.71

6. Challenge Submission and Conclusions

Our challenge submission for all three languages consisted of the output from both the IITM-C and IITM-CR based multilingual pooled models described in Section 4 on their respective language evaluation set. The Gujarati results from the IITM-C based pooled model placed in the leader-board for that language.

In this paper we have described the ISI submission for the Low Resource Speech Recognition Challenge for Indian Languages. Our best system, a *single* multilingual LSTM-CTC based ASR system across three languages, shows for the first time the viability of such models on sub-100 hour training sets.

7. Acknowledgements

The research is based upon work supported by the Office of the Director of National Intelligence (ODNI), Intelligence Advanced Research Projects Activity (IARPA), via AFRL Contract #FA8650-17-C-9116.

The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of the ODNI, IARPA, or the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright annotation thereon.

8. References

[1] A. Parlikar, S. Sitaram, A. Wilkinson, and A. W. Black, "The Festvox Indic Frontend for Grapheme-to-Phoneme Conversion," in *3rd Workshop on Indian Language Data: Resources and Evaluation*, 2016.

[2] "IndicTTS: Common Label Set: Indian Language

speech sound label set (ILSL12) version 2.1.6," https://www.iitm.ac.in/donlab/tts/downloads/cls/cls_v2.1.6.pdf.

[3] "Machine Translation for English Retrieval of Information in Any Language (MATERIAL) program," https://www.iarpa.gov/index.php/research-programs/material.

[4] L. Besacier, E. Barnard, A. Karpov, and T. Schultz, "Automatic speech recognition for under-resourced languages: A survey," *Speech Commun.*, vol. 56, pp. 85–100, Jan. 2014.

[5] N. Jaitly and G. E. Hinton, "Vocal Tract Length Perturbation (VTLP) improves speech recognition," in *Proceedings of the International Conference on Machine Learning (ICML) 2013 Workshop on Deep Learning for Audio, Speech and Language Processing, Atlanta, Georgia, USA, June 16-21, 2013*.

[6] Z. Tüske, P. Golik, D. Nolden, R. Schlüter, and H. Ney, "Data augmentation, feature combination, and multilingual neural networks to improve ASR and KWS performance for low-resource languages," in *INTERSPEECH 2014, 15th Annual Conference of the International Speech Communication Association, Singapore, September 14-18, 2014*, pp. 1420–1424.

[7] A. Ragni, K. M. Knill, S. P. Rath, and M. J. F. Gales, "Data augmentation for low resource languages," in *INTERSPEECH 2014, 15th Annual Conference of the International Speech Communication Association, Singapore, September 14-18, 2014*, pp. 810–814.

[8] X. Cui, V. Goel, and B. Kingsbury, "Data augmentation for deep convolutional neural network acoustic modeling," in *2015 IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2015, South Brisbane, Queensland, Australia, April 19-24, 2015*, pp. 4545–4549.

[9] T. Ko, V. Peddinti, D. Povey, and S. Khudanpur, "Audio augmentation for speech recognition," in *INTERSPEECH 2015, 16th Annual Conference of the International Speech Communication Association, Dresden, Germany, September 6-10, 2015*, pp. 3586–3589.

[10] G. Zavaliagkos, M. Siu, T. Colthurst, and J. Billa, "Using untranscribed training data to improve performance," in *The 5th International Conference on Spoken Language Processing, Sydney, Australia, 1998*. ISCA, 1998.

[11] K. Yu, M. J. F. Gales, L. Wang, and P. C. Woodland, "Unsupervised training and directed manual transcription for LVCSR," *Speech Communication*, vol. 52, no. 7-8, pp. 652–663, 2010.

[12] S. Thomas, M. L. Seltzer, K. Church, and H. Hermansky, "Deep neural network features and semi-supervised training for low resource speech recognition," in *IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2013, Vancouver, BC, Canada, May 26-31, 2013*, pp. 6704–6708.

[13] S. Thomas, S. Ganapathy, and H. Hermansky, "Multilingual MLP features for low-resource LVCSR systems," in *2012 IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2012, Kyoto, Japan, March 25-30, 2012*. IEEE, 2012, pp. 4269–4272.

[14] J. Cui, B. Kingsbury, B. Ramabhadran, A. Sethy *et al.*, "Multilingual representations for low resource speech recognition and keyword search," in *2015 IEEE Workshop on Automatic Speech Recognition and Understanding, ASRU 2015, Scottsdale, AZ, USA, December 13-17, 2015*. IEEE, 2015, pp. 259–266.

[15] P. Swietojanski, A. Ghoshal, and S. Renals, "Unsupervised cross-lingual knowledge transfer in DNN-based LVCSR," in *2012 IEEE Spoken Language Technology Workshop (SLT), Miami, FL, USA, December 2-5, 2012*. IEEE, 2012, pp. 246–251.

[16] J. Huang, J. Li, D. Yu, L. Deng, and Y. Gong, "Cross-language knowledge transfer using multilingual deep neural network with shared hidden layers," in *IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2013, Vancouver, BC, Canada, May 26-31, 2013*, pp. 7304–7308.

[17] R. Caruana, "Multitask learning," *Mach. Learn.*, vol. 28, no. 1, pp. 41–75, Jul. 1997.

[18] D. Chen, B. Mak, C. Leung, and S. Sivadas, "Joint acoustic modeling of triphones and trigraphemes by multi-task learning deep neural networks for low-resource speech recognition," in *IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2014, Florence, Italy, May 4-9, 2014*, pp. 5592–5596.

[19] H. Lin, L. Deng, D. Yu, Y. Gong *et al.*, "A study on multilingual acoustic modeling for large vocabulary ASR," in *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing, ICASSP 2009, 19-24 April 2009, Taipei, Taiwan*. IEEE, 2009, pp. 4333–4336.

[20] G. Heigold, V. Vanhoucke, A. W. Senior, P. Nguyen *et al.*, "Multilingual acoustic models using distributed deep neural networks," in *IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2013, Vancouver, BC, Canada, May 26-31, 2013*, pp. 8619–8623.

[21] N. T. Vu, D. Imseng, D. Povey, P. Motlícek *et al.*, "Multilingual deep neural network based acoustic modeling for rapid language adaptation," in *IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2014, Florence, Italy, May 4-9, 2014*, pp. 7639–7643.

[22] T. Schultz and A. Waibel, "Fast bootstrapping of LVCSR systems with multilingual phoneme sets," in *Fifth European Conference on Speech Communication and Technology, EUROSPEECH 1997, Rhodes, Greece, September 22-25, 1997*, G. Kokkinakis, N. Fakotakis, and E. Dermatas, Eds. ISCA, 1997.

[23] A. Graves, S. Fernández, F. J. Gomez, and J. Schmidhuber, "Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks," in *Machine Learning, Proceedings of the Twenty-Third International Conference (ICML 2006), Pittsburgh, Pennsylvania, USA, June 25-29, 2006*, pp. 369–376.

[24] D. Amodei, R. Anubhai, E. Battenberg, C. Case *et al.*, "Deep Speech 2 : End-to-End Speech Recognition in English and Mandarin," in *Proceedings of the 33nd International Conference on Machine Learning, ICML 2016, New York City, NY, USA, June 19-24, 2016*, pp. 173–182.

[25] H. Soltau, H. Liao, and H. Sak, "Neural speech recognizer: Acoustic-to-word LSTM model for large vocabulary speech recognition," *CoRR*, vol. abs/1610.09975, 2016.

[26] D. Povey, V. Peddinti, D. Galvez, P. Ghahremani *et al.*, "Purely sequence-trained neural networks for ASR based on lattice-free MMI," in *Interspeech 2016, 17th Annual Conference of the International Speech Communication Association, San Francisco, CA, USA, September 8-12, 2016*, pp. 2751–2755.

[27] J. Billa, "Improving LSTM-CTC based ASR performance in domains with limited training data," *CoRR*, vol. abs/1707.00722, 2017.

[28] ——, "Dropout approaches for LSTM based speech recognition systems," in *Proceedings of the 2018 IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP '18, Calgary, Canada, April 15-20, 2018*.

[29] S. Tong, P. N. Garner, and H. Bourlard, "Multilingual Training and Cross-lingual Adaptation on CTC-based Acoustic Model," *CoRR*, vol. abs/1711.10025, 2017.

[30] Y. Miao, M. Gowayyed, and F. Metze, "EESEN: End-to-end speech recognition using deep RNN models and WFST-based decoding," in *2015 IEEE Workshop on Automatic Speech Recognition and Understanding, ASRU 2015, Scottsdale, AZ, USA, December 13-17, 2015*, pp. 167–174.

[31] M. Schuster and K. K. Paliwal, "Bidirectional recurrent neural networks," *IEEE Trans. Signal Processing*, vol. 45, no. 11, pp. 2673–2681, 1997.

[32] S. Semeniuta, A. Severyn, and E. Barth, "Recurrent dropout without memory loss," in *COLING 2016, 26th International Conference on Computational Linguistics, Proceedings of the Conference: Technical Papers, December 11-16, 2016, Osaka, Japan*, 2016, pp. 1757–1766.