Interspeech 2022
18-22 September 2022, Incheon, Korea

Multi-Task End-to-End Model for Telugu Dialect and Speech Recognition

Aditya Yadavalli, Ganesh S Mirishkar, Anil Kumar Vuppala

Speech Processing Laboratory
International Institute of Information Technology, Hyderabad
Gachibowli, Hyderabad, Telangana, 500032

{aditya.yadavalli, mirishkar.ganesh}@research.iiit.ac.in, anil.vuppala@iiit.ac.in

Abstract

Conventional Automatic Speech Recognition (ASR) systems are susceptible to dialect variations within a language, thereby adversely affecting the ASR. Therefore, the current practice is to use dialect-specific ASRs. However, dialect-specific information or data is hard to obtain making it difficult to build dialect-specific ASRs. Furthermore, it is cumbersome to maintain multiple dialect-specific ASR systems for each language. We build a unified multi-dialect End-to-End ASR that removes the need for a dialect recognition block and the need to maintain multiple dialect-specific ASRs for three Telugu regional dialects: Telangana, Coastal Andhra, and Rayalaseema. We find that pooling the data and training a multi-dialect ASR benefits the low-resource dialect the most – an improvement of over 9.71% in relative Word Error Rate (WER). Subsequently, we experiment with multi-task ASRs where the primary task is to transcribe the audio and the secondary task is to predict the dialect. We do this by adding a Dialect ID to the output targets. Such a model outperforms naive multi-dialect ASRs by up to 8.24% in relative WER. Additionally, we test this model on a dialect recognition task and find that it outperforms strong baselines by 6.14% in accuracy.

Index Terms: multi-dialect, speech recognition, sequence-to-sequence, dialect recognition

1. Introduction

With the advent of neural networks in the recent years, the performance of Automatic Speech Recognition (ASR) under various scenarios has improved tremendously. Despite these advancements in spoken language technologies, dialect or accent variation is still a major challenge. This is because a considerable portion of variation in dialect can be seen at multiple linguistic levels: phonology, lexicon, syntax, and occasionally in semantics. [1] use Principal Component Analysis (PCA) as a statistical tool to measure speaker variability. They report that accent contributes second most to speaker variability after gender. Furthermore, [2] show that ASRs do not perform well when evaluated on a dialect of the same language that was not used in the training phase. Thus, state-of-art ASR services such as Google, use dialect-specific ASRs.[1]

Conventional multi-dialect ASR systems are outperformed by dialect-specific ASRs [3]. However, multi-dialect ASR is an attractive solution as it is difficult to obtain dialect-specific data or dialect related information. Moreover, it is also easier to maintain a single ASR model for all the dialects of a language. Motivated by these reasons, a number of methods have been proposed to build an ASR that closes the gap between multi-dialect and dialect-specific ASR's performance. [4] explore whether using dialect-specific pronunciation dictionary can improve the performance of a German ASR. [5] compare grapheme-based Recurrent Neural Network-Connectionist Temporal Classification (RNN-CTC) to phoneme-based RNN-CTC models. They report that grapheme-based models perform better than phoneme-based models under multi-dialect English conditions. Additionally, they study the performance of grapheme-based ASRs when phoneme recognition is given as an auxiliary task. They report that such a model outperforms a grapheme-based RNN-CTC by comparing both on multiple English dialects. [6, 7] use a model where it jointly learns an accent classifier and a multi-accent acoustic model and report significant improvements in WER. [8] study the performance of End-to-End ASRs by employing different accent embeddings extracted from speech into their model.

Our work is most closely related to [3, 9, 10]. [3], among other things, propose a Multi-Task Learning (MTL) Acoustic Model (AM), where predicting context-dependent (CD) phones is the primary task and predicting the dialect is the secondary task. Our work differs from this in two ways: Firstly, we work with End-to-End models where AM is not optimised modularly but along with pronunciation model and Language Model (LM). [9] propose various methods to build a unified multi-dialect Listen Attend Spell (LAS) ASR [11]. Among them, they propose a multi-task learning framework where they add a Dialect ID to the text as output targets. While we follow a similar approach in building our ASR model, our work is different from theirs in two ways: the Transformer-based End-to-End architecture (see Section 3) we use is different from the LSTM-based LAS architecture that they use, we conduct our experiments on spontaneous speech corpus whereas they use a conversational speech corpus, which presents its own set of challenges. In addition to these differences, we test our multi-task ASR models on its secondary task (dialect recognition), whereas [3, 9] report only their ASR performance. [10] use MTL, where there is a shared Transformer-based encoder to predict the accent and transcribe the speech. In this paper, we take a simpler and a more elegant approach where both our encoder and decoder are jointly optimised to predict the dialect and transcribe the speech.

Our language of interest in this paper is Telugu. It is a South-Central Dravidian language spoken by a large population in two states of India: Telangana and Andhra Pradesh. A noticeable amount of the population in these two states cannot read or write as evidenced by low literacy rates. This leaves them with only one form of communication – the spoken form. Therefore, building spoken language technologies for this community would be of utmost importance. So far, research on Telugu ASRs has largely focused on improving the performance of the "standardised" version of the language [13, 14]. While [15] collect dialect-specific Telugu data, they do not report their

[1] For example, https://cloud.google.com/speech-to-text/docs/languages has 16 different ASRs for different Arabic dialects.

10.21437/Interspeech.2022-10739

Dialect	Example Sentence
Coastal Andhra	prawi pOrudu otu wappaka veyAlaMdi
Rayalaseema	mAku mA palleVtUru aMte cAnA iRtamu
Telangana	gA PutbAl guriMci ayiwe nAku masw gA weVlusu rA bE

Table 1: *Example Sentences of Various Dialects From the Corpus in WX notation [12].*

ASR performance individually on regional dialects.

With these motivations, we make the following contributions in this paper:

- We build a unified multi-dialect End-to-End ASR model for three geographically close Telugu dialects: Coastal Andhra, Telangana, and Rayalaseema. We find that adding a Dialect ID to the output targets leads to an improvement of upto 13.92% in relative WER over our baseline dialect-specific ASRs.
- Since our model jointly predicts the Dialect ID and transcribes audio, we also test it on dialect recognition task. We find that our model achieves near perfect results in certain cases and outperforms strong baselines by 6.14%.

2. Dataset

We train all our ASR models on approximately 1200 hours of crowd-sourced spontaneous speech data, part of which has been described by [15]. This includes data from three regional dialects: Coastal Andhra, Rayalaseema, and Telangana. The test set of each dialect is about 8 hours of speech. All the audio used in this study is 16KHz with 16-bit encoding. As the speakers who contributed to the data were asked to speak about a particular topic, we were able to ensure that there is no domain mismatch in the dialect-specific data. This allows us to study the role dialects play in an ASR system better. We also ensure that there are no overlapping utterances in our train and test sets. We present the statistics of the dataset in Table 2.

Dialect	Coastal Andhra	Telangana	Rayalaseema
Train	70.8K	84.8K	65.3K
Test	1.9K	2K	1.9K

Table 2: *Number of utterances per dialect for training and test set (K for thousand)*

2.1. Telugu Dialects

Telugu dialects can be broadly categorised into three following regional dialects: Coastal Andhra, Telangana, and Rayalaseema. These dialects are formed primarily due to the culture of the area, and influence of other languages spoken in the neighboring states [16]. The Mughals and Nizams ruled over the Telangana region, whose official languages were Persian and Urdu. Thus, we notice the influence of Urdu with many nativised Urdu words present in Telangana [17]. Here are some such examples: kAka, jAgA, xavAKAnA.[2] Coastal Andhra dialect is influenced by Sanskrit and English due to historical reasons [18]. Finally, the Rayalaseema dialect is influenced by Tamil and Kannada due to geographical reasons [18]. Table 1 shows a sample sentence of each dialect from the corpus.

[2]We use WX notation to represent all Telugu words and sentences in this paper [12]

2.2. Analysis

Even though the dataset used in paper is from crowd-sourced spontaneous speech corpus, our manual verification of the data revealed that a number of urban speakers who did not speak in their regional Telugu accent but in the "standardised" Telugu dialect. However, they were found to be using vocabulary from their dialect. Therefore, to avoid this confound, we limit ourselves to textual analysis in this section.

We choose to fine-tune *IndicBERT* [19] on dialect classification task. *IndicBERT* is an ALBERT [20] based pre-trained multilingual model trained on one of the largest publicly available corpora on Indian languages, *IndicCorp* [19]. It achieves state-of-the-art performance on many Indic benchmarks.

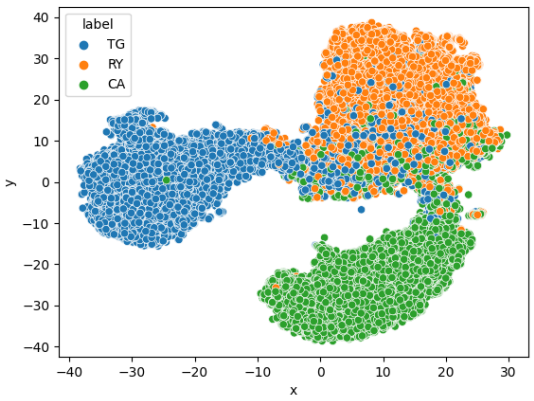

Figure 1: *t-SNE plot of IndicBERT sentence representations of the three dialects. In this plot, TG is Telangana, CA is Coastal Andhra, and RY is Rayalaseema.*

Figure 1 shows the t-SNE [21] plot of the sentence representations of the *IndicBERT* model after fine-tuning on dialect classification task. Rayalaseema representations form a cluster and there is very little deviation and overlap with the main clusters of other dialects. On the other hand, both Coastal Andhra and Telangana have significant overlaps with Rayalaseema's main cluster. This indicates that it is tougher to classify Coastal Andhra and Telangana dialects correctly than Rayalaseema.

3. Experimental Setup

We conduct all our experiments using the ESPnet toolkit [22]. The input to the End-to-End model are 80-dimensional log-mel features extracted on the fly. Previously, Conformer has been shown to achieve state-of-the-art results on many standard

Experiment	Example Sentence
`Dialect-Specific`	prawi pOrudu otu wappaka veyAlaMdi
`Multi-Dialect`	prawi pOrudu otu wappaka veyAlaMdi
`Multi-Dialect + Dialect ID`	prawi pOrudu otu wappaka veyAlaMdi $\langle CA \rangle$
`Dialect ID + Multi-Dialect`	$\langle CA \rangle$ prawi pOrudu otu wappaka veyAlaMdi

Table 3: *Example Sentences of Coastal Andhra Dialect of Various Speech Recognition Models*

datasets [23]. Therefore, we choose to use a 12 layer Conformer encoder with 8 attention heads and a 6 layer Transformer [24] decoder with 4 attention heads to train our ASR model. We train both the encoder and decoder with a drop rate of 0.1 for a maximum of 40 epochs. We take these hyperparameters from the `Librispeech recipe` [3] which is tuned to perform well on a dataset sized similar to the one used in this paper and make two changes: the initial learning rate used is 5×10^{-4}, and we use early stopping with the criterion to stop training if the model's performance on the validation set does not improve for 10 consecutive epochs. The models are trained based on Hybrid CTC/Attention [25] architecture. The training is done within a Multi-Task Learning framework and the learning objective (L_{MTL}) is as follows:

$$L_{MTL} = \lambda \log p_{ctc}(c|x) + (1 - \lambda) \log p^*_{att}(c|x)$$

We use this objective as [25] not only show that such an architecture gives modest improvements over the Attention model but also show that the CTC loss term helps the Attention model converge faster. In the above equation, λ is the multi-task coefficient which should satisfy the following: $0 \leq \lambda \leq 1$. We found that λ set to 0.3 while training and 0.4 when decoding to give us the best results on the validation set. c is the output target. In our study, these are subword units. [26] show that using subwords over characters leads to better performance of End-to-End ASRs. On the other hand, using subwords is preferred over words as it handles Out of Vocabulary (OOV%) well, which is a huge challenge in an agglutinative language like Telugu. We use SentencePiece [27] to split the words into its corresponding subwords. We ensure that Dialect ID (see Section 4) is not split into subwords by SentencePiece. We find that 500 tokens for dialect-specific ASR models and 1200 tokens for multi-dialect models give us the best performance. Finally, x is the input acoustic features.

We also train an independent 16-block Transformer-based LM with an embedding size of 128 and a hidden encoder size of 512 for 25 epochs. The decoder uses this LM with a weight of 0.6 to predict the output targets.[4] This method of integrating LM into the ASR is called Shallow Fusion [28] and it is shown to give superior results compared to other forms of LM fusion [29]. [5] We use a beam search with size 10 to predict the output sequence. Our experiments are conducted using four Nvidia Geforce GTX 1080Ti GPUs.

[3] `https://github.com/espnet/espnet/blob/master/egs2/librispeech/asr1/conf/tuning/train_asr_conformer.yaml`

[4] The rest of the weight is given to the Hybrid CTC/Attention model.

[5] Only the training transcripts are used to train the LM as well as the SentencePiece tokeniser

4. Results & Discussion

In this section, we present both the performance of the ASRs and dialect recognition of the multi-task ASRs.[6]

4.1. Speech Recognition

[3] show that dialect-specific conventional ASRs outperform multi-dialect conventional ASRs. Therefore, we build dialect-specific ASR models and treat them as baselines in this paper. However, our aim is to build a single unified multi-dialect ASR as it is difficult to obtain dialect-specific information or data. Thus, we build a naive multi-dialect ASR by pooling all the dialect-specific data. Subsequently, we build multi-task ASRs where the primary task is to transcribe the audio and the secondary task is to predict the dialect. We do this by incorporating dialect-specific information into a unified multi-dialect ASR in the training phase. [30] build a language independent multilingual ASR. They experiment by adding a language ID to the beginning of an utterance and report significant improvement in their ASR's performance. On the other hand, [9] report that adding an ID to the end of the utterance results in a better performing ASR. Therefore, we experiment with both kinds of ASRs. More specifically, when we refer to `Multi-Dialect + Dialect ID`, we mean a model where Dialect ID is added at the end of the utterance and when we refer to `Dialect ID + Multi-Dialect`, we mean a model where Dialect ID is added in the beginning of the utterance. An example utterance for each of these models is provided in Table 3.

The results for all the four ASR models explained above are presented in Table 4. It can be observed that the multi-task ASRs, on average, outperform dialect-specific and naive multi-dialect ASR that is built by pooling all the dialect-specific data. This shows that adding a Dialect ID helps the ASR model learn the dialect biases better. Out of the two multi-task ASRs, `Dialect ID + Multi-Dialect` model, on average, performs the best. It can also be observed that it outperforms dialect-specific model by 3.47% and 3.37% absolute WER when tested on Coastal Andhra and Rayalaseema test sets respectively. We observe a marginal improvement of upto 0.4% when Dialect ID is added as opposed to when it is not in multi-dialect ASR when evaluated on Telangana. However, dialect-specific Telangana model outperforms all the Telangana models. This shows that a multi-dialect system for a high-resource dialect is not as beneficial as it is for a low-resource dialect.

Unlike what [9] observe with their LAS multi-dialect ASRs, we find that adding the Dialect ID before the utterance helps the ASR model perform better. Previously, such an observation has been when tackling other problems such as multilingual ASRs and multilingual neural machine translation [30, 31]. This improvement when Dialect ID is added in front of the utterance as opposed to when it is appended could be explained

[6] When we refer to multi-task ASR, we mean Dialect ID + Multi-Dialect ASR and Multi-Dialect ASR + Dialect ID ASR

Experiment	Coastal Andhra	Telangana	Rayalaseema	Average	Training Time
`Dialect-Specific`	34.3	**25.4**	24.2	27.96	9 days, 17 hours, 57 mins
`Multi-Dialect`	33.6	27.0	21.85	27.53	9 days, 7 hours, 33 mins
`Multi-Dialect + Dialect ID`	31.75	26.6	21.61	26.69	**7 days, 7 hours, 8 mins**
`Dialect ID + Multi-Dialect`	**30.83**	26.99	**20.83**	**26.20**	7 days, 11 hours, 20 mins

Table 4: *WER(%) of Various Speech Recognition Models*

by the auto-regressive nature of Transformers – the architecture used in our decoder. The decoder has the opportunity to depend on the right prediction of the Dialect ID to transcribe the audio when the Dialect ID is added to the beginning of the utterance. However, this is not the case when the Dialect ID is appended to the utterance. From Table 4, we can also observe that the multi-task ASR models converge, upto 21.84%, faster than `Multi-Dialect` model. This indicates that explicitly giving the multi-dialect ASR model dialect-specific information not only helps the model perform better but also converge faster. We plan to conduct a much more thorough analysis on convergence rates in the future.

4.2. Dialect Recognition

Figure 2 shows the confusion matrix of dialect recognition of `Multi-Dialect + Dialect ID` model. The diagonals of the matrix correspond to dialect recognition accuracies while the other cells correspond to error rates. We find that the model performs almost perfectly in predicting Rayalaseema dialect. On the other hand, the model misclassifies Coastal Andhra dialect as Rayalaseema 5% of the time. It can also be observed from the t-SNE plot (see Figure 1) that a lot of the Coastal Andhra samples overlap with the Rayalaseema cluster, indicating that Coastal Andhra dialect has a significant overlap with Rayalaseema dialect. On the other hand, the dialect recognition accuracy of `Dialect ID + Multi-Dialect` is 78.5%. The model's decoder only has access to the audio to recognise the dialect as opposed `Multi-Dialect + Dialect ID`'s decoder. Since, as previously explained, we use a spontaneous speech dataset where a noticeable number of urban speakers do not speak in the native accent, it is difficult for a model which only has access to audio to predict the right dialect.

To better understand the difficulty of dialect recognition task on our dataset, we consider two popular BERT-based Indic-specific pre-trained models – *IndicBERT* [19] and *MuRIL Base* [32] – that achieve state-of-the-art results on many Indic benchmarks. *MuRIL* is a BERT [33] based model trained with Masked Language Modeling and Translation Language Modeling objectives [34]. It is trained on 17 Indic languages (1) monolingual data: Common Crawl OSCAR corpus [7] and Wikipedia [8] (2) Translated Data: PMIndia [35] and other in-house parallel corpus (3) Transliterated Data: Dakshina Dataset [36] and transliterated Wikipedia data using *indic-trans* [37]. We fine-tune these two pre-trained models on the same training transcripts provided to the ASR models. We tokenise the input sequence using the respective model's pre-trained tokeniser. We append a classification head to the pre-trained models. An initial learning of 1×10^{-5} is used with Adam optimiser [38]. We train these models for 10 epochs. They achieve an accuracy of 88.58% and 90.94% respectively. Inspite of these models having access to a lot more Telugu text, and Indic text in general, our multi-task ASR model outperforms these models by over 6%. This shows the importance of having access to both audio and text for dialect recognition.

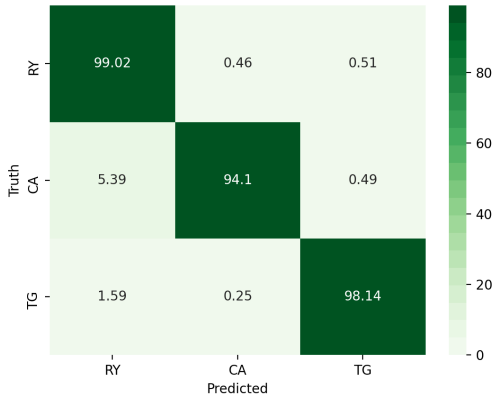

Figure 2: *Dialect Recognition Confusion Matrix of* `Multi-Dialect + Dialect ID`*. Here, TG is Telangana, CA is Coastal Andhra, and RY is Rayalaseema.*

5. Conclusion

In this paper, we study the effectiveness of a multi-task ASR that jointly predicts the dialect and transcribes the audio. We find that it outperforms a naive multi-dialect ASR, which is built by pooling all the dialect specific data, by upto 8.24% relative WER (see Coastal Andhra performance in Table 4). Additionally, the multi-task ASR model outperforms the dialect-specific ASR model by upto 13.92% relative WER (see Rayalaseema performance in Table 4). Furthermore, we evaluate our multi-task ASR on dialect recognition task and find that it performs relatively well with an accuracy of 97.08%.

In this study, we make modifications to the training transcripts of the ASR to incorporate dialect-specific information in a multi-dialect ASR. In the future, we would like to propose novel modifications to incorporate dialect-specific information into multi-dialect ASR at two levels: features and the model architecture. We hope that our future work would further improve the performance of multi-dialect ASR, making it an attractive solution over maintaining multiple dialect-specific ASRs.

6. References

[1] C. Huang, T. Chen, S. Li, E. Chang, and J.-L. Zhou, "Analysis of speaker variability," in *INTERSPEECH*, 2001.

[2] F. Biadsy, P. J. Moreno, and M. Jansche, "Google's cross-dialect

[7] `https://oscar-corpus.com`

[8] `https://www.tensorflow.org/datasets/catalog/wikipedia`

arabic voice search," in *2012 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2012, pp. 4441–4444.

[3] M. Elfeky, M. Bastani, X. Velez, P. Moreno, and A. Waters, "Towards acoustic model unification across dialects," in *2016 IEEE Spoken Language Technology Workshop (SLT)*, 2016, pp. 624–628.

[4] N. Beringer, F. Schiel, and P. Regel-Brietzmann, "German regional variants - a problem for automatic speech recognition?" in *Proc. 5th International Conference on Spoken Language Processing (ICSLP 1998)*, 1998, p. paper 0201.

[5] K. Rao and H. Sak, "Multi-accent speech recognition with hierarchical grapheme based models," in *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2017, pp. 4815–4819.

[6] X. Yang, K. Audhkhasi, A. Rosenberg, S. Thomas, B. Ramabhadran, and M. A. Hasegawa-Johnson, "Joint modeling of accents and acoustics for multi-accent speech recognition," *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 1–5, 2018.

[7] A. Jain, M. Upreti, and P. Jyothi, "Improved accented speech recognition using accent embeddings and multi-task learning," in *INTERSPEECH*, 2018.

[8] T. Viglino, P. Motlícek, and M. Cernak, "End-to-end accented speech recognition," in *INTERSPEECH*, 2019.

[9] B. Li, T. N. Sainath, K. C. Sim, M. Bacchiani, E. Weinstein, P. Nguyen, Z. Chen, Y.-Q. Wu, and K. Rao, "Multi-dialect speech recognition with a single sequence-to-sequence model," *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 4749–4753, 2018.

[10] J. Zhang, Y. Peng, V. T. Pham, H. Xu, H. Huang, and C. E. Siong, "E2e-based multi-task learning approach to joint speech and accent recognition," in *Interspeech*, 2021.

[11] 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.

[12] R. Gupta, P. Goyal, and S. Diwakar, "Transliteration among indian languages using wx notation," in *KONVENS*, 2010.

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

[14] V. V. R. Vegesna, K. Gurugubelli, H. K. Vydana, B. Pulugandla, M. Shrivastava, and A. K. Vuppala, "Dnn-hmm acoustic modeling for large vocabulary telugu speech recognition," in *MIKE*, 2017.

[15] G. S. Mirishkar, M. D. Naroju, S. Maity, P. Yalla, A. K. Vuppala *et al.*, "Cstd-telugu corpus: Crowd-sourced approach for large-scale speech data collection," in *2021 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC)*. IEEE, 2021, pp. 511–517.

[16] K. Mannepalli, P. N. Sastry, and M. Suman, "Mfcc-gmm based accent recognition system for telugu speech signals," *International Journal of Speech Technology*, vol. 19, no. 1, pp. 87–93, 2016.

[17] V. Ithagani, "Linguistic convergence and divergence in telugu-urdu contact situation: A study with special reference to telangana dialect." 2014.

[18] S. Shivaprasad and M. Sadanandam, "Identification of regional dialects of telugu language using text independent speech processing models," *International Journal of Speech Technology*, vol. 23, no. 2, pp. 251–258, 2020.

[19] D. Kakwani, A. Kunchukuttan, S. Golla, G. N.C., A. Bhattacharyya, M. M. Khapra, and P. Kumar, "IndicNLPSuite: Monolingual corpora, evaluation benchmarks and pre-trained multilingual language models for Indian languages," in *Findings of the Association for Computational Linguistics: EMNLP 2020*. Online: Association for Computational Linguistics, Nov. 2020, pp. 4948–4961.

[20] Z. Lan, M. Chen, S. Goodman, K. Gimpel, P. Sharma, and R. Soricut, "Albert: A lite bert for self-supervised learning of language representations," *ArXiv*, vol. abs/1909.11942, 2020.

[21] L. van der Maaten and G. E. Hinton, "Visualizing data using t-sne," *Journal of Machine Learning Research*, vol. 9, pp. 2579–2605, 2008.

[22] S. Watanabe, T. Hori, S. Karita, T. Hayashi, J. Nishitoba, Y. Unno, N. Enrique Yalta Soplin, J. Heymann, M. Wiesner, N. Chen, A. Renduchintala, and T. Ochiai, "ESPnet: End-to-end speech processing toolkit," in *Proceedings of Interspeech*, 2018, pp. 2207–2211.

[23] A. Gulati, J. Qin, C.-C. Chiu, N. Parmar, Y. Zhang, J. Yu, W. Han, S. Wang, Z. Zhang, Y. Wu, and R. Pang, "Conformer: Convolution-augmented transformer for speech recognition," 2020.

[24] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," 2017.

[25] S. Kim, T. Hori, and S. Watanabe, "Joint ctc-attention based end-to-end speech recognition using multi-task learning," *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 4835–4839, 2017.

[26] C.-C. Chiu, T. N. Sainath, Y. Wu, R. Prabhavalkar, P. Nguyen, Z. Chen, A. Kannan, R. J. Weiss, K. Rao, K. Gonina, N. Jaitly, B. Li, J. Chorowski, and M. Bacchiani, "State-of-the-art speech recognition with sequence-to-sequence models," *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 4774–4778, 2018.

[27] T. Kudo and J. Richardson, "Sentencepiece: A simple and language independent subword tokenizer and detokenizer for neural text processing," in *EMNLP*, 2018.

[28] A. Kannan, Y. Wu, P. Nguyen, T. N. Sainath, Z. Chen, and R. Prabhavalkar, "An analysis of incorporating an external language model into a sequence-to-sequence model," *ICASSP*, 2018.

[29] S. Toshniwal, A. Kannan, C.-C. Chiu, Y. Wu, T. N. Sainath, and K. Livescu, "A comparison of techniques for language model integration in encoder-decoder speech recognition," in *2018 IEEE Spoken Language Technology Workshop (SLT)*, 2018, pp. 369–375.

[30] S. Watanabe, T. Hori, and J. R. Hershey, "Language independent end-to-end architecture for joint language identification and speech recognition," in *2017 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, 2017, pp. 265–271.

[31] M. Johnson, M. Schuster, Q. V. Le, M. Krikun, Y. Wu, Z. Chen, N. Thorat, F. Viégas, M. Wattenberg, G. Corrado *et al.*, "Google's multilingual neural machine translation system: Enabling zero-shot translation," *Transactions of the Association for Computational Linguistics*, vol. 5, pp. 339–351, 2017.

[32] S. Khanuja, D. Bansal, S. Mehtani, S. Khosla, A. Dey, B. Gopalan, D. K. Margam, P. Aggarwal, R. T. Nagipogu, S. Dave, S. Gupta, S. C. B. Gali, V. Subramanian, and P. P. Talukdar, "Muril: Multilingual representations for indian languages," *ArXiv*, vol. abs/2103.10730, 2021.

[33] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," in *NAACL*, 2019.

[34] G. Lample and A. Conneau, "Cross-lingual language model pretraining," in *NeurIPS*, 2019.

[35] B. Haddow and F. Kirefu, "Pmindia - a collection of parallel corpora of languages of india," *ArXiv*, vol. abs/2001.09907, 2020.

[36] B. Roark, L. Wolf-Sonkin, C. Kirov, S. J. Mielke, C. Johny, I. Demirsahin, and K. Hall, "Processing south asian languages written in the latin script: the dakshina dataset," 2020.

[37] I. A. Bhat, V. Mujadia, A. Tammewar, R. A. Bhat, and M. Shrivastava, "Iiit-h system submission for fire2014 shared task on transliterated search," in *FIRE '14*, 2014.

[38] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *CoRR*, vol. abs/1412.6980, 2015.