Interspeech 2022
18-22 September 2022, Incheon, Korea

An Improved Transformer Transducer Architecture for Hindi-English Code Switched Speech Recognition

Ansen Antony, Sumanth Reddy Kota, Akhilesh Lade, Spoorthy. V, Shashidhar G. Koolagudi*

Department of Computer Science and Engineering, National Institute of Technology, Karnataka, Surathkal, India

{ansen.181co107, sumanthreddy.181co225, akhileshlade.181co130, koolagudi}@nitk.edu.in, vspoorthy036@gmail.com

Abstract

Due to the extensive usage of technology in many languages throughout the world, interest in Automatic Speech Recognition (ASR) systems for Code-Switching (CS) in speech has grown in recent years. Several studies have shown that End-to-End (E2E) ASR is easier to adopt and works much better in monolingual settings. E2E systems are likewise widely recognised for requiring massive quantities of labelled speech data. Since there is a scarcity in the availability of large amount of CS speech, E2E ASR takes longer computation time and does not offer promising results. In this work, an E2E ASR model system using a transformer-transducer architecture is introduced for code-switched Hindi-English speech, and also addressed training data scarcity by leveraging the vastly available monolingual data. Specifically, the language-specific modules in the Transformer are pre-trained by leveraging the vastly available single language speech datasets. The proposed method also provides a Word Error Rate (WER) of 29.63% and Transliterated Word Error Rate (T-WER) of 27.42% which is better than the state-of-the-art by 2.19% .

Index Terms: Transformer, Automatic Speech Recognition, Transducer, Code-Switching

1. Introduction

Code-Switching (CS) refers to the phenomenon where a multilingual speaker switches between two or more languages. Parsing, Machine Translation, Automatic Speech Recognition (ASR), Information Retrieval & Information Extraction, and semantic processing are the significant hurdles in Code-Switched speech processing. When an input is combined with two or more languages, traditional approaches trained using a single language do not provide noteworthy results. Even for problems that are believed to be addressed, performance degrades according to the number and level of mixed-languages present [1].

Multiple studies have been conducted to date on building ASR systems for CS speech. At the beginning, Deep Neural Networks (DNNs) constituted a crucial component of traditional hybrid ASR systems. Consistent with certain research studies, these models achieve better results in many circumstances with less training data, although their training often needs strong context-dependent trees. Despite this, End-To-End (E2E) techniques are becoming more popular as the training process becomes less complicated. E2E systems are trained to translate a sequence of input features to a series of characters and for hybrid systems the outputs of a DNN are trained using state probabilities from Hidden Markov Model [2, 3]. Additionally, the flexibility of intermediary modeling simplifies the construction of an ASR model. E2E ASR systems are significantly easier and more efficient to utilize than hybrid ASR systems [4]. E2E ASR systems are becoming more common since they do not require explicit alignments and require more minor hyperparameter adjustments. However, this simple architecture of E2E ASR systems still requires a large quantity of training data. When the datasets are not large enough, E2E ASR systems perform poorly [5].

Being successful in CS ASR systems is one of the main priorities in the ASR domain for the moment, and it is a challenging task. To attain state-of-the-art performance, several attempts have been performed in this arena. Despite these substantial advancements, language systems are still unable to handle CS data as efficiently as monolingual data. When the input speech is mixed with several languages, typical ASR systems tend to break down, and the rate of performance loss is proportional to the quantity of code-switched speech present. However, most commercial CS ASR systems are built to identify many languages, limiting the application's usage. A Single Language ASR uses a standard Speech-Transformer consisting of a single audio and label encoder, whereas a multi-encoder-decoder Transformer that learns the characteristics of each distinct language present in the code-switched speech and effectively segregate among these distinct languages present in the speech [6].

MUCS 2021: Multilingual and code-switching ASR challenges for low resource Indian languages proposes a Hindi-English code-switching ASR system. The ASR systems were designed with an E2E code-switching paradigm in mind, the architecture describes a Transformer-based hybrid Connectionist Temporal Classification (CTC) attention model. We propose a multi-encoder-decoder Transformer Transducer (TT) model in this paper that understands individual language characteristics and effectively discriminates among the languages mixed in the target speech. In the decoder module, we propose two symmetric language-specific encoders. Long-term temporal dependencies in speech data can be better modeled with transformers. The Transformer's language-specific modules can be pre-trained using the publicly available huge single language speech datasets [7]. In order to compensate for the lack of CS training data, the proposed architecture incorporates the pre-training of both label and audio encoders. Compared to the baseline Transformer, which has a single-encoder system, and the Kaldi-based state-of-the-art hybrid ASR system, this proposed architecture outperforms both of them.

The rest of the paper is organized as follows. In section 2, we present the overview of the various components of the TT architecture. Section 3 consists of the details of the experimental setup, parameters of the ASR model, the model description. Section 4 presents the results obtained from the proposed architectures. And the paper is concluded in Section 5.

10.21437/Interspeech.2022-10763

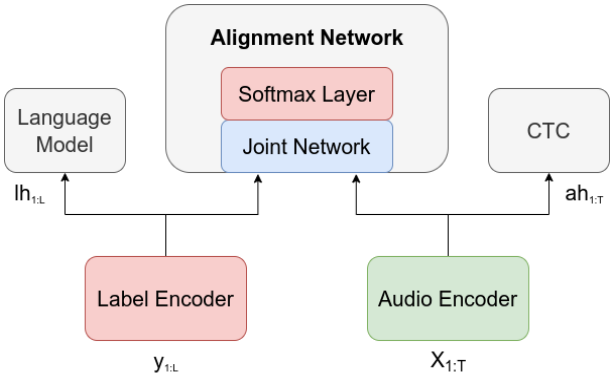

Figure 1: *Transformer-Transducer Model [6]*

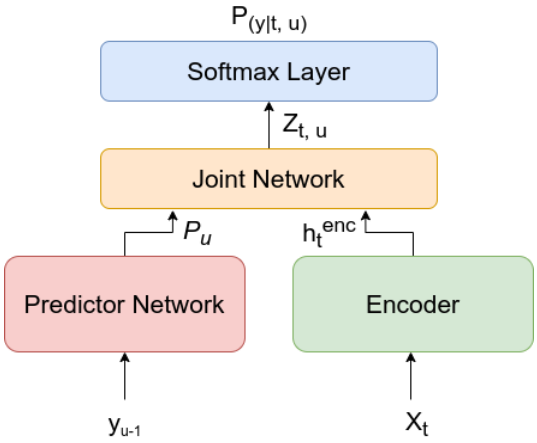

Figure 2: *RNN-Transducer [13]*

2. Transformer Transducer Model

In this paper, we propose a TT model for CS in speech containing Hindi and English languages. The model uses transformer computation blocks as encoders and decoders and the model is trained with transducer loss (RNN-T loss). Hence the name Transformer-Transducer. [8]. Figure 1 illustrates the TT model.

2.1. Transducer Architecture and Loss (RNN-Transduer)

Transducer models are broadly used for online speech recognition as a result of their ability to do real-time streaming speech recognition [9] and low memory footprint [10]. The transducer loss is an alignment-based loss that in encoder-decoder models perform the job equivalent of cross-attention, making it useful for closed captioning generation [11]. Since the emergence of self-attention-based transformer models and data augmentation technologies like SpecAugment, transducers have also provided competitive performance to attention-based encoder-decoder models. This procedure necessitates a fine-tuning step, which can be time-consuming caused by the numerous choices that must be made in neural-network optimization, such as learning rate, optimizer, and so on. Figure 2 illustrates the transducer loss function. RNN-Transducer uses both audio encoding at timestamp "t" (x_t) and transcript history generated so far ($y_{(t-1)}$) to generate probability distribution over output units. The RNN-transducer model has 3 modules encoder, predictor and joint network. The encoder extracts the acoustic features of audio, and the predictor works as language model to learn the language from the training data. The joint network takes in the input from encoder and predictor and from these two it creates a label. A feature vector x_t of n-dimensions, which is extracted at timestamp "t" and is passed to the encoder. It provides to the h_t encoder the hidden representation of last layers output at that particular timestamp. $y_{(t-1)}$ is the previous prediction(character excluding the blanks) goes as an input to prediction network which gives P_u as an output. h_t and h_{pred} goes as inputs to the joint network where the outputs from both predictor network and encoder are combined and gives $Z_{t,u}$. This passes through the softmax layer which produces the probability distribution over output units($P_{(y|t,u)}$) [12].

2.2. Transformer

Transformer is an attention-based encoder-decoder architecture wherein the encoder translates an input sequence into an abstract continuous representation. The decoder then takes that continuous representation and, while being fed the previous output, creates a single output step by step [14].

The transformer model architecture, as illustrated in Figure 3, where the input is sent into the input embedding layer for word embeddings, where the learnt vector representation of each word is looked up. The embeddings are then injected with positional information since, unlike recurrent neural networks, the transformer encoder has no recurrence, therefore some information about the locations must be provided to the input embeddings. The data is now sent to the encoder layer, which consists of two sub-modules: multi-headed attention and a completely linked network with residual connections surrounding both sublayers, followed by layer normalization.

To map every word in the feed to other words, the encoder employs a specific attention mechanism. Two multi-headed attention layers, a pointwise feed-forward layer, residual connections, and layer normalisation are among the characteristics. A sub-layer similar to that of the encoder exists in the decoder. The decoder's job is to use a linear classifier and a softmax to create text sequences.

The decoder is autoregressive and starts with a start token [15]. It takes a list of previous outputs as well as encoder outputs, including information about the input's attention. The decoder stops decoding when it generates a token as an output. Transformers use the attention mechanism to make better predictions. They have the potential to be more efficient, particularly when encoding or producing large sequences.

2.3. Custom masked training for the languages

We suggest two adjustments to the transducer network's training method to address the issues faced by transducer models such as the alignment network and the label encoder, which must learn alignments for multiple languages and know when to switch between languages for next word prediction.

Masking target tokens randomly in the label encoder: The alignment network's attention to learning audio alignments that are not reliant on the target token is achieved with the help of the mask tag. The label encoder in the conversational speech is more resistant to utterance anomalies like disfluency and non-speech noises with the help of the mask tags.

Whenever there is a language switch, add an LID tag to the target sequence: The LID tags assists the label encoder in switching languages within a single speech. With the help of LID tags, the alignment network can understand on which

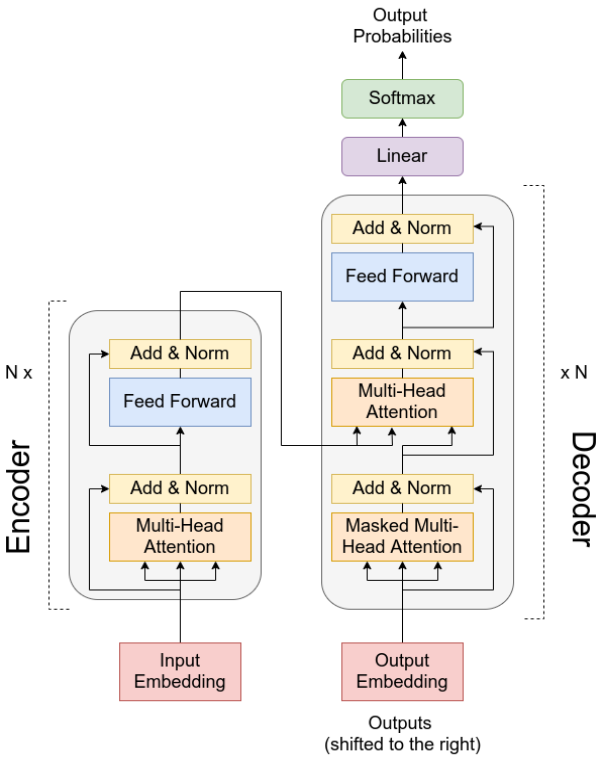

Figure 3: *Transformer model architecture [14]*

language it is working at the moment.

A <mask> token is used to replace every masked word. For the LID tags, we use the <en> or <hin> tags to indicate the beginning of an English or Hindi word respectively.

2.4. Multi-label and Multi audio Encoder

In a typical E2E ASR, the audio and labels are encoded by one single Encoder. This model propose a Multi-Label/Multi-Audio Encoder that utilizes monolingual data and uses a learnable sigmoid gate to combine their information before sending it forward to the joint network, as shown in Figure 4. This contributes to the learning of particular language characteristics as well as the successful discrimination of languages blended in the target speech. In the decoder module, the suggested design comprises of two language-specific encoders which are symmetric and multi-head attention blocks which are language-specific. We also intend to overcome the insufficiency of code-switched training data by pre-training each of the Transformer's language-specific modules using vastly available single language voice dataset, utilizing transfer learning methods [6].

3. Experiments and Results

3.1. Dataset

In this proposed model, MUCS 2021 Challenge Dataset which is publicly available, is used [16]. It is a Hindi-English code-switched dataset extracted from spoken tutorials and these tutorials contain various domains of technical topics in computer science, with transcripts including mathematical symbols and other technical content. The training data consists of speech from 520 speakers and the test data contains 30 disjoint speakers. The range of vocabulary and technical aspects increase with

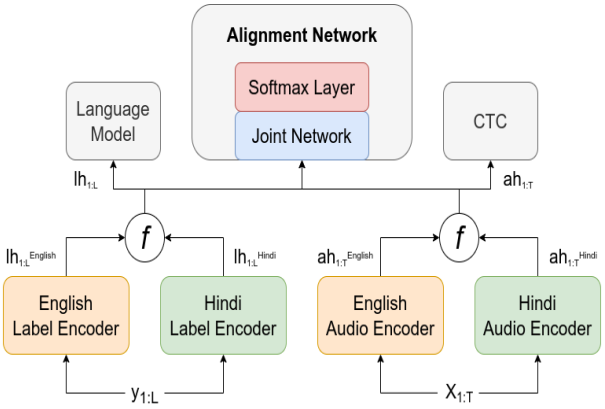

Figure 4: *Transformer Tradnsucer Model with Multi-Label/ Multi Audio Encoder*

the increasing level of the tutorials. The sentence timestamps in the segments file were used to derive segments from the audio file to match with transcripts in the text file. The audio files in the dataset are sampled at 16kHz, and stored with 16 bit encoding. The overlap between the test and train dataset is 33.9% and blind test and train overlap is 2.9% respectively. The percentage of Out-of-vocabulary (OOV) words encountered in test and blind-test for Hindi-English subtask is 12.5% and 19.6%. The training set consists of 44250 unique utterances over ∼90 hours. The test and blind test datasets (a part of the dataset set aside, which is never used before, and used to test the model after all training iterations and hyperparameter tuning are complete) consist of 2900 and 3830 utterances for over ∼5 hours and ∼6.5 hours respectively. The dataset used for pre-training the encoders are taken mozilla common voice and it has English language with 81,085 voices that sums up to 2,224 validated hours of speech with 71% of speakers with age below 40 years and remaining with speakers above age 40 which comprises of 54% females and remaining are males and Hindi language with 299 voices that sums up to 21 validated hours of speech with 62% of speakers with age below 40 and remaining with age above 40 which comprises 74% males and remaining with females.

3.2. Environment Setup

All the experiments proposed in this paper are implemented and presented on ESPnet speech processing toolkit [17] version 0.9.7. SpecAugment is applied for data augmentation during the pre-training phase of the model and also the the fine-tuning stages. All models for the experimentation are carried on a NVIDIA dgx1 machine with 1 Tesla p100 GPU of memory 16gb and Dual 20-core Intel® Xeon® E5-2698 v4 2.2 GHz CPU with 28672 NVIDIA cuda cores.

Table 1: *Description of CS dataset*

	Train	**Test**	**Blind**
Size (Hours)	90	5	6.50
Unique sentences	44250	2900	3830
Speakers	520	30	35
Vocab (Words)	17850	3215	3530

3.3. Transformer Transducer

In the first configuration of the TT model, the architecture consists of an encoder network consisting of 12 layers, each layer consists of 2048 units and a dropout rate of 0.1. The architecture also has a decoder network consisting of 6 layers, each layer consists of 2048 units and a dropout rate of 0.1. Each layer has twelve 64-dimensional heads which are concatenated to form an attention vector of 512 dimensions. The model was trained with forty epochs. The early stopping patience is 0 and the noam optimizer is used with a learning rate set to 10. To allow adaptive optimisers to compute correct statistics of the gradients, 8000 warmup steps are used. Decoding is done with a beam search size of 5 with score norm transducer. The TT model trained comparatively faster (around 1 day) than the hybrid CTC model (around 3 days), with nearly the same parameters.

The second configuration of the TT model has some minor changes over the first configuration to improve performance without degrading the ASR accuracy. This architecture consists of an encoder network consisting of 18 layers, each layer consists of 2048 units and a dropout rate of 0.3. The architecture also has a decoder network consisting of 9 layers, each layer consists of 2048 units and a dropout rate of 0.3. The model was trained with forty epochs. The early stopping patience is 0.3 .

The third configuration of the TT model has the same configurations as the second one except, in this we add a VGG layer (transformer-enc-input-layer: vgg2l) with causal convolution and introduced masked training and language ID tags for CS moments. The VGG layer is added to incorporate contextual information into the Transformer networks.

3.4. Transformer Transducer with Multi-Encoders

Similar to the TT architecture, the multi-encoder-decoder architecture also has transformers as encoder and transducers as decoders. The main difference is in the encoders. We use 2 encoders, one is trained on English and the other on Hindi language. The monolingual datasets used to pre-train the encoders are from the publicly available mozilla commonvoice datasets. Both the encoders are incorporated to learn the individual language attributes and are joined to a single decoder. The architecture consists of a dual array transformer VGG2L encoder network consisting of 12 layers, each layer consists of 768 units and a decoder dropout rate of 0.1 and attention dropout rate of 0.4. The architecture also has a decoder network consisting of 6 layers, each layer consists of 768 units and a decoder dropout rate of 0.1 and attention dropout rate of 0.4. Each layer has twelve 64-dimensional heads which are concatenated to form an attention vector of 512 dimensions.

3.5. Transformer Transducer with Pre-trained Multi-Encoders

In the pretrained encoders TT model, we use a pretrained transformer encoder with a transducer decoder. The encoder is trained on mozilla commonvoice datasets. Each entry in the dataset has a unique MP3 file and accompanying text file.

The dataset currently consists of 2,185 hours in the English language and 16 hours in the Hindi language. This architecture has the same configurations mentioned in subsection 3.4.

When encoders are pre-trained with individual languages it makes the model more accurate to train as the data we use for training is code switched and it helps the custom ID tags to differentiate between Hindi and English.

3.6. Results

We compare the results of the TT model with the existing state-of-the-art Hybrid DNN-HMM ASR models. In the first set of experiments, we look into different iterative configurations on the TT model. We also compute the Transliterated-Word-Error-Rate (T-WER) along with the standard WER. If an English word is correctly predicted in English or in its transliterated version in Hindi, T-WER will count it as properly predicted in the reference text. Table 2 shows the WER performance of multiple TT configurations. We observe that the TT model presents a WER of 31.05%, better than the existing DNN-HMM baseline model [8].

We also explore the performance changes when minor optimizations are made to the existing model. Adding a VGG layer(configuration 2) into the existing encoder-decoder model presents a slightly smaller improvement in the WER. But, we see a marginal improvement of 1.40% WER when compared with the DNN-HMM model. Overall, the improvement in the performance reduces the WER from the 32.45% to 31.05%.

Next, we run the proposed multi-label encoder-decoder ASR model with a view to improve the performance and accuracy of the WER of Hindi-English mixed speech via monolingual data. The WER is determined by the total sum of the Hindi and English words which were predicted incorrectly. The results of the models along with the improvements on the WER through the proposed TT models are shown in Table 2. From the results, it is seen that the monolingual language encoders present a better rendering of encoded speech and also provides a cleaner representation on the respective languages [6]. The Multi-label Encoder-Decoder model achieved a 30.26% WER which is a 2.19% improvement over the baseline model. Similarly, we run the pre-trained multi-encoder decoder model achieved a 29.63% WER. The multi-label encoders clearly depict better results over the TT models. But, this model lacks the ability to overcome the language boundaries of the acoustic space. This could be resolved by training the models on multiple set of speakers with different acoustic representations [18].

Table 2: *WERs for Hindi-English Code-switched Speech*

Model	**WER (%)**	**T-WER (%)**
DNN-HMM model	32.45	29.80
TT Configuration 1	38.36	36.71
TT Configuration 2	31.18	30.54
TT Configuration 3	31.05	29.88
TT with Multi-Encoders	30.26	29.37
TT with Pre-trained Multi-Encoders	29.63	27.42

4. Conclusion

In this paper, we presented the details of the dataset and a TT ASR model for code-switched speech. We trained the model with 100+ hours of code-switched Hindi English dataset. We also pre-trained the encoders with largely and publicly available mozilla common voice Hindi and English speech corpus. The TT model with pre-trained multi-encoders showed promise by producing a 29.63% WER which is 2.19% better than the present state of the art code swtiched asr model which has 32.45% WER.

5. References

[1] S. Sitaram, K. R. Chandu, S. K. Rallabandi, and A. W. Black, "A survey of code-switched speech and language processing," *Computing Research Repository (CoRR)*, vol. abs/1904.00784, pp. 1–70, 2019.

[2] D. Wang, X. Wang, and S. Lv, "An overview of end-to-end automatic speech recognition," *Symmetry*, vol. 11, p. 1018, 08 2019.

[3] K. Audhkhasi, A. Rosenberg, G. Saon, A. Sethy, B. Ramabhadran, S. Chen, and M. Picheny, "Recent progress in deep end-to-end models for spoken language processing," *IBM Journal of Research and Development*, vol. 61, pp. 1–10, 07 2017.

[4] J. Perero-Codosero, F. M. Espinoza-Cuadros, and L. Gómez, "A comparison of hybrid and end-to-end ASR systems for the IberSpeech-RTVE 2020 speech-to-text transcription challenge," *Applied Sciences*, vol. 12, p. 903, 01 2022.

[5] N. T. Vu, D.-C. Lyu, J. Weiner, D. Telaar, T. Schlippe, F. Blaicher, E.-S. Chng, T. Schultz, and H. Li, "A first speech recognition system for Mandarin-English code-switch conversational speech," in *International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2012, pp. 4889–4892.

[6] X. Zhou, E. Yılmaz, Y. Long, Y. Li, and H. Li, "Multi-encoder-decoder transformer for code-switching speech recognition," 10 2020, pp. 1042–1046.

[7] A. Zeyer, P. Bahar, K. Irie, R. Schlüter, and H. Ney, "A comparison of transformer and lstm encoder decoder models for asr," in *Automatic Speech Recognition and Understanding Workshop (ASRU)*, 2019, pp. 8–15.

[8] Q. Zhang, H. Lu, H. Sak, A. Tripathi, E. McDermott, S. Koo, and S. Kumar, "Transformer transducer: A streamable speech recognition model with transformer encoders and rnn-t loss," in *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2020, pp. 7829–7833.

[9] X. Chen, Y. Wu, Z. Wang, S. Liu, and J. Li, "Developing real-time streaming transformer transducer for speech recognition on large-scale dataset," *International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 5904–5908, 2021.

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

[11] A. Zeyer, A. Merboldt, R. Schlüter, and H. Ney, "A new training pipeline for an improved neural transducer," *Interspeech 2020*, pp. 2812–2816, 2020.

[12] S. Wang, P. Zhou, W. Chen, J. Jia, and L. Xie, "Exploring RNN-transducer for chinese speech recognition," *CoRR*, vol. abs/1811.05097, 2018.

[13] Q. Zhang, H. Lu, H. Sak, A. Tripathi, E. McDermott, S. Koo, and S. Kumar, "Transformer transducer: A streamable speech recognition model with transformer encoders and RNN-T loss," *International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 7829–7833, 2020.

[14] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in Neural Information Processing Systems*, 2017, pp. 5998–6008.

[15] Z. Tian, J. Yi, J. Tao, Y. Bai, S. Zhang, and Z. Wen, "Spike-triggered non-autoregressive transformer for end-to-end speech recognition," *ArXiv*, vol. abs/2005.07903, 2020.

[16] A. Diwan, R. Vaideeswaran, S. Shah, A. Singh, S. R. K. M., S. Khare, V. Unni, S. Vyas, A. Rajpuria, C. Yarra, A. R. Mittal, P. K. Ghosh, P. Jyothi, K. Bali, V. Seshadri, S. Sitaram, S. Bharadwaj, J. Nanavati, R. Nanavati, K. Sankaranarayanan, T. Seeram, and B. Abraham, "Multilingual and code-switching ASR challenges for low resource indian languages," *Computing Research Repository (CoRR)*, vol. abs/2104.00235, 2021.

[17] S. Watanabe, T. Hori, S. Karita, T. Hayashi, J. Nishitoba, Y. Unno, N. E. Y. Soplin, J. Heymann, M. Wiesner, N. Chen, A. Renduchintala, and T. Ochiai, "ESPnet: End-to-end speech processing toolkit," *Computing Research Repository (CoRR)*, vol. abs/1804.00015, 2018.

[18] Y. Miao, H. Zhang, and F. Metze, "Speaker adaptive training of deep neural network acoustic models using I-vectors," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 23, pp. 1938–1949, 11 2015.