Dual Script E2E Framework for Multilingual and Code-Switching ASR

Mari Ganesh Kumar, Jom Kuriakose, Anand Thyagachandran, Arun Kumar A*, Ashish Seth*, Lodagala V S V Durga Prasad*, Saish Jaiswal*, Anusha Prakash, and Hema A Murthy

Indian Institute of Technology Madras
mari@cse.iitm.ac.in (MGK), jom@cse.iitm.ac.in (JK), anushaprakash@smail.iitm.ac.in (AP), hema@cse.iitm.ac.in (HAM)

Abstract

India is a multilingual country, having 22 official languages and many more mother tongues [1]. Building automatic speech recognition (ASR) systems for Indian languages is a challenging problem owing to the low resource setting. Moreover, due to globalization and migration, code-switching is a common phenomenon. Code-switching is the alternation between multiple languages in a single conversation. English is one of the most widely used languages in code-switching. The two major goals of the Indic ASR challenge 2021 [2, 3] are the advancement of multilingual and code-switching ASR for Indian languages. The Indic ASR challenge 2021 consists of two sub-tasks. The main aim of this paper is to build multilingual ASR by pooling data from multiple low-resource languages. We propose novel end-to-end (E2E) ASR systems using an in-house common label set (CLS) representation, which has been successful in text-to-speech (TTS) synthesis for low resource Indian languages [4, 5]. The proposed models are evaluated as part of the Indic ASR Challenge 2021.

A major issue with building ASR systems for Indian languages in a multilingual context is that the languages have different grapheme representations. This difference in the grapheme representation limits the ASR models in learning common sounds across languages. Although Indian languages have different written scripts, they share a common phonetic base. Inspired by this, a common label set for 13 Indian languages was developed for TTS synthesis [6]. In [7], a rule-based unified parser for 13 Indian languages was proposed to convert grapheme-based Indian language text to phoneme-based CLS. CLS has shown to be successful for TTS systems developed using both hidden Markov model (HMM) based approaches [6], recent end-to-end frameworks [4, 5]. For training multilingual ASR systems, recent studies use the phone-based CLS representation obtained using the rule-based unified parser [7] for training multilingual ASR systems.

This paper uses the CLS representation in two different ways to train multilingual E2E transformer models. First, we convert the native script of all Indian languages to the CLS representation and then train the E2E transformer model using the CLS representation. This model takes advantage of the same sounds across different languages and maps them together. This improves the performance of the system on low resource languages. However, the model decodes the audio files only in CLS. We propose novel language identification (LID) and machine transliteration techniques to recover the native script after decoding. This system achieves 5% improvement over the baseline system for the multilingual task on the challenge development data.

In the second system, we propose integrating the LID and machine transliteration backend within the E2E transformer model itself. We train this model using both the CLS representation and the native language script simultaneously. During decoding, the CLS output is discarded, and the model outputs the Indian language script directly. This system achieves a 6% improvement over the baseline for the multilingual task on the challenge development data. We also show our results on the code-switching task of the challenge.

The remainder of the paper is organised as follows. The Indic ASR challenge and the associated datasets are briefly described in Section 2. All the ASR systems proposed in this paper use the CLS transcriptions for training. The conversion of the native language script to the CLS representation is described in Section 3. The proposed systems are explained in Section 4. Results are presented and discussed in Sections 5 and 6 respectively. The work is concluded in Section 7.

2. Indic ASR Challenge 2021

The Indic ASR challenge 2021 consists of two sub-tasks. In sub-task 1, the main objective is to build a multilingual ASR system for Indian languages. The dataset for sub-task 1 consists of six low resource Indian languages, namely, Hindi, Marathi,
The unified parser is a rule-based grapheme to phoneme converter designed for Indian languages. The unified parser takes a UTF-8 word as input and recognizes the language-specific phones are given separate labels. The Indian languages are mapped together and given a common representation obtained using the unified parser (see Table 2). Instead of the native language script, the CLS E2E model uses a common representation as proposed in [13]. An example of the word “action”, its parsed output in CMU phones and the final CLS representation is given in Table 2.

### 4. Multilingual ASR Systems

The audio files were first downsampled to 8000Hz, and 80 mel filter bank energies along with pitch were extracted to be used as features. The paper uses three different E2E models (1 baseline, 2 proposed) for building multilingual ASR. The three variants of the E2E model are discussed in this section. ESPNet toolkit [14] was used to train the E2E models. The models were trained using Hybrid CTC-attention [14][15] based transformer architectures.

#### 4.1. Baseline E2E Model

This simple baseline model pools data from all the languages during training to build a multilingual system. Since different Indian languages use different scripts, this model does not take advantage of the common sounds present across all the languages.

#### 4.2. CLS E2E Model

Instead of the native language script, the CLS E2E model uses a common representation obtained using the unified parser (see Section 3). Since this model is trained using CLS, it decodes back only in CLS labels. It is to be noted that the native text (in UTF-8) to CLS mapping is not one-to-one due to rules such as schwa deletion, geminate correction, and syllable parsing [13]. Due to these different rules, one-to-one mapping of CLS back to the native script of the right language is challenging. Examples of such confusions are given in Table 3.

Since the native script cannot be retrieved from the CLS using a one-to-one mapping, machine transliteration models are trained and used. Machine translation techniques used for language translation are extended to transliteration by inserting space between each character in a word and considering each character as a word. Neural machine transliteration models are trained using the default configuration of open neural machine translation (ONMT) toolkit [16] to learn the character level mapping from CLS to the native text. A unimodal naive Bayes classifier using TF-IDF features obtained from decoded CLS is used to identify the language.
Baseline E2E Model

CTC layer

Transformer

Decoder

Transformer: Encoder

Output [in Native Script]

filterbank coefficients with pitch

B.

CLS E2E Model

Output [in CLS]

Transformer: Encoder

CTC layer

Transformer

Decoder

filterbank coefficients with pitch

C.

Dual Script E2E Model

Output [in Native Script]

Output 2 [ in CLS]

Transformer: Encoder

CTC layer 2

Transformer

Decoder 2

filterbank coefficients with pitch

Figure 1: Illustration of proposed E2E hybrid CTC-attention models using CLS

Table 3: Confusions in CLS to native script mapping

| Language  | CLS  | Possible mappings |
|-----------|------|-------------------|
| Hindi     | kAmor | कामोर | कामोर |
| Bengali   | Sid|Ant | সিড|অন্ট |

Table 4: Detailed transformer architecture used for E2E acoustic models

| Name          | Value | Name          | Value | Name          | Value |
|---------------|-------|---------------|-------|---------------|-------|
| Attention     | 256   | Encoder       | 12    | Decoder       | 6     |
| Attention     | 4     | Encoder       | 2048  | Decoder       | 2048  |
| Output BPU    | 5000  | Output 2 BPU | 800   |               |       |

4.3. Dual Script E2E Model

In this system, we integrate the LID and machine transliteration backend within the E2E model by adding additional decoders. We use two CTC layers and decoders to predict the CLS and native language script (in UTF-8) simultaneously. This modified hybrid CTC-attention model is trained over a loss function, giving equal weight to predicting both the CLS and native language script. This simultaneous training is expected to cue the E2E model with phonemes that are common across languages. During decoding, the CTC layer and the decoder corresponding to CLS are discarded, and the model outputs the Indian language script directly. Figure 1 illustrates the differences between the three E2E models trained in this work.

4.4. Implementation Details

For all the three different E2E models mentioned above, the same network architecture was used. Table 4 presents the network architecture in detail. These E2E models were trained with both byte-pair sub-word units (BPU) [17] and character units (CU). For sub-task 1, the E2E models were trained only using the corresponding sub-task data for 20 epochs. For sub-task 2, the models obtained in sub-task 1 were fine-tuned for 20 epochs. No external data was used for training acoustic models.

Additionally, for sub-task 1, a transformer-based language model was trained using external text data. Text data from Indic TTS [18], and AI4Bharat NLP corpora [19] were used to train the language model. The transformer-based [11] language model was trained for 1 epoch on the combined text corpora with about 150 million sentences pooled from all six languages used in sub-task 1. No language model was used for sub-task 2.

For the CLS E2E model, long short term memory (LSTM) based encoder-decoder model with global attention was used for neural machine transliteration at the word level (see Section 4.2). The machine transliteration models were trained language-wise. Since the character vocabulary of CLS to the native script is limited, the system learns transliteration. When trained with the IndicTTS text data [18] and the ASR Challenge train text, this model achieved 1.78% WER and 0.44% character error rate (CER) on sub-task 1 development data, averaged across all six languages.

For the LID part of the CLS E2E model, TF-IDF features were used. Term-frequency (TF) gives more importance to a more frequent word in a document, whereas inverse-document frequency (IDF) tries to penalize a word if it occurs in multiple documents. Multi-gram TF-IDF features were extracted at character and word levels from each sentence, and a simple Bayes classifier was used to predict the language. This system was also trained with IndicTTS text data [18] and ASR Challenge train text, and achieved an accuracy of 99.7% on sub-task 1 development data.

For sub-task 1, HMM-based and time delay neural network (TDNN) based models were provided as baseline models. For sub-task 2, in addition to these models, an E2E conformer [20] model was also used as a baseline. The details of these baseline models can be found in [3].

5. Results

5.1. Results of sub-task 1

Table 5 shows the results of all the models discussed in this paper on the development data for sub-task 1. The dual script system, which uses the character units (System H), outperforms the best baseline (System B) by 1% average WER, without using any language model. With a language model, this system (System L) outperforms the best baseline by 6% average WER.

The results of the final systems submitted to the challenge on the blind test set are given in Table 6. In contrast to results in Table 5 for the blind data, the baseline system gave better results than the proposed systems (in Avg-1, computed from all 6 languages). However, it is notable that, without Marathi, the average WER (Avg-2) improved by 6% over the best baseline for the dual script system (System L). These results are discussed in detail in Section 6.1.
In this paper, we propose two novel multilingual ASR systems. From the results in Table 7, it is observed that E2E models give better results when trained with CLS instead of just pooling the native language script. This observation is common to both E2E systems trained with byte-pair and character units. This observation verifies our conjecture that CLS cues the E2E model in learning common sounds across languages, thus enabling faster convergence in a low resource setting.

In Tables 5 and 7, we see that the dual script system consistently outperforms the CLS system with a separate backend. However, in the blind set of sub-task 1 (Table 6), we observe that the CLS system with a separate backend generalises better, especially for Marathi. The CLS model with a separate backend is able to scale better mainly because of the very few unique sentences in the training data for Marathi (see Table 1) and usage of more text data to train the backend (see Section 4). We also note that the Indic ASR challenge website mentions that the channel encoding scheme between the Marathi blind and non-blind (train & test) datasets are different.

### 6.2. Byte-pair and character units for low resource languages

From the results in Table 5 it is observed that the byte-pair system consistently gives better results for Tamil, Telugu and Gujarati. In comparison, the character level system achieves better results for Hindi, Marathi and Odia. This is mainly because the training data of Hindi, Marathi and Odia has many repeated sentences. This has led to poor byte-pair estimation compared to the other three languages, which have about 30,000 unique sentences.

### 6.3. Limitations

Although sub-task 1 is monolingual data, we observe that Marathi and Odia have many English words in the train and development sets, with very few unique and short sentences (see Table 1). Also, the code-switched data in sub-task 2 is very noisy. It contains English words in the Indian language script with many additional characters such as Greek letters.  

7. Conclusion

Given the low resource setting and multiple grapheme representations of Indian languages, training ASR systems is challenging. The use of CLS for training multilingual E2E ASR systems ensures that common sounds are mapped together, leading to better ASR models. We also propose a dual script architecture trained simultaneously on CLS and native language text. This model can be used with a rule-based many-to-one mapping for different scripts. The model learns the native script and also gets the advantage of the common labels. Results across various models also show that the choice of basic units (CU or BPU) largely depends on the available training data.

8. Acknowledgements

This work was carried out as a part of the following projects– “Natural Language Translation Mission” (cs2021072/MEIT003119) and “Speech to Speech Machine Translation” (cs2021152/PSA003119), funded by the Ministry of Electronics and Information Technology (MeIT) and the Office of the Principal Scientific Advisor (PSA) to the Government of India, respectively.
9. References

[1] D. P. Pattanayak, Multilingualism in India. Multilingual Matters, 1990, no. 61.

[2] “Multilingual and code-switching asr challenges for low resource indian languages,” https://navana-tech.github.io/IS21SS-indicASRchallenge/ 02 2021, (Accessed 2/4/2021 0:20).

[3] A. Diwan, R. Vaideeswaran, S. Shah, A. Singh, S. Raghavan, S. Khare, V. Unni, S. Vyas, A. Rajput, C. Yarra, A. Mittal, P. K. Ghosh, P. Jyothi, K. Bali, V. Seshadr, S. Sitaram, S. Bharadwaj, J. Nanavati, R. Nanavati, K. Sankaranarayanan, S. Teeram, and B. Abraham, “Multilingual and code-switching asr challenges for low resource indian languages,” arXiv preprint arXiv:2104.00235, 2021.

[4] A. Prakash, A. Leela Thomas, S. Umesh, and H. A. Murthy, “Building Multilingual End-to-End Speech Synthesizers for Indian Languages,” in 10th ISCA Speech Synthesis Workshop (SSW), 2019, pp. 194–199.

[5] A. Prakash and H. A. Murthy, “Generic Indic Text-to-Speech Synthesizers with Rapid Adaptation in an End-to-End Framework,” in Interspeech, 2020, pp. 2962–2966.

[6] B. Ramani, S. Lilly Christina, G. Anushiya Rachel, V. Sherlin Solomi, M. K. Nandwana, A. Prakash, S. Aswin Shannmugam, R. Krishnan, S. Khisore, K. Samudrivijaya, P. Vijayalakshmi, T. Nagarajan, and H. A. Murthy, “A common attribute based unified HTS framework for speech synthesis in Indian languages,” in Speech Synthesis Workshop (SSW), 2013, pp. 291–296.

[7] A. Baby, N. Nishanthi, A. L. Thomas, and H. A. Murthy, “A unified parser for developing Indian language text to speech synthesizers,” in International Conference on Text, Speech, and Dialogue. Springer, 2016, pp. 514–521.

[8] A. Datta, B. Ramabhadran, J. Ennond, A. Kannan, and B. Roark, “Language-agnostic multilingual modeling,” in ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). IEEE, 2020, pp. 8239–8243.

[9] S. Thomas, K. Audhkhasi, and B. Kingsbury, “Transliteration based data augmentation for training multilingual asr acoustic models in low resource settings,” Proc. Interspeech 2020, pp. 4736–4740, 2020.

[10] V. M. Shetty and S. Umesh, “Exploring the use of Common Label Set to Improve Speech Recognition of Low Resource Indian Languages,” IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 2021 (Accepted for publication), accepted for publication.

[11] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, “Attention is all you need,” arXiv preprint arXiv:1706.03762, 2017.

[12] K. Park and J. Kim, “g2p.e,” https://github.com/Kyubyong/g2p, 2019.

[13] A. L. Thomas, A. Prakash, A. Baby, and H. A. Murthy, “Code-switching in Indic Speech Synthesisers,” in INTERSPEECH, 2018, pp. 1948–1952.

[14] S. Watanabe, T. Hori, S. Karita, T. Hayashi, J. Nishitoba, Y. Unno, N. Enrique Yalta Soplín, 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. [Online]. Available: http://dx.doi.org/10.21437/ Interspeech.2018-1456.

[15] S. Watanabe, T. Hori, S. Kim, J. R. Hershey, and T. Hayashi. “Hybrid etc/attention architecture for end-to-end speech recognition,” IEEE Journal of Selected Topics in Signal Processing, vol. 11, no. 8, pp. 1240–1253, 2017.

[16] G. Klein, Y. Kim, Y. Deng, J. Senellart, and A. M. Rush, “Opennmt: Open-source toolkit for neural machine translation,” arXiv preprint arXiv:1701.02810, 2017.

[17] T. Kado, “Subword regularization: Improving neural network translation models with multiple subword candidates,” 2018.

[18] A. Baby, A. L. Thomas, N. N. L., and H. A. Murthy, “Resources for Indian languages,” in Community-based Building of Language Resources (International Conference on Text, Speech and Dialogue), 2016, pp. 37–43.

[19] D. Kakwani, A. Kunchukuttan, S. Golla, G. N.C., A. Bhattacharya, M. M. Khapra, and P. Kumar, “IndicNLPSuite: Monolingual Corpora, Evaluation Benchmarks and Pre-trained Multilingual Language Models for Indian Languages,” in Findings of EMNLP, 2020.

[20] A. Gulati, J. Qin, C.-C. Chiu, N. Parmar, Y. Zhang, J. Yu, W. Han, S. Wang, Z. Zhang, Y. Wu et al., “Conformer: Convolution-augmented transformer for speech recognition,” arXiv preprint arXiv:2005.08100, 2020.