INTERSPEECH 2021
30 August – 3 September, 2021, Brno, Czechia

Dual Script E2E Framework for Multilingual and Code-Switching ASR

Mari Ganesh Kumar[1], *Jom Kuriakose*[2], *Anand Thyagachandran*, *Arun Kumar A*[*], *Ashish Seth*[*], *Lodagala V S V Durga Prasad*[*], *Saish Jaiswal*[*], *Anusha Prakash*[3], *Hema A Murthy*[4]

Indian Institute of Technology Madras, India
[1] mari@cse.iitm.ac.in, [2] jom@cse.iitm.ac.in, [3] anushaprakash@smail.iitm.ac.in,
[4] hema@cse.iitm.ac.in

Abstract

India is home to multiple languages, and training automatic speech recognition (ASR) systems is challenging. Over time, each language has adopted words from other languages, such as English, leading to code-mixing. Most Indian languages also have their own unique scripts, which poses a major limitation in training multilingual and code-switching ASR systems.

Inspired by results in text-to-speech synthesis, in this paper, we use an in-house rule-based phoneme-level common label set (CLS) representation to train multilingual and code-switching ASR for Indian languages. We propose two end-to-end (E2E) ASR systems. In the first system, the E2E model is trained on the CLS representation, and we use a novel data-driven backend to recover the native language script. In the second system, we propose a modification to the E2E model, wherein the CLS representation and the native language characters are used simultaneously for training. We show our results on the multilingual and code-switching (MUCS) ASR challenge 2021. Our best results achieve $\approx 6\%$ and 5% improvement in word error rate over the baseline system for the multilingual and code-switching tasks, respectively, on the challenge development data.

Index Terms: speech recognition, low-resource, multilingual, common label set, dual script

1. Introduction

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 MUCS 2021 challenge [2, 3] are the advancement of multilingual and code-switching ASR for Indian languages.

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 MUCS 2021 Challenge [2, 3].

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], and recent end-to-end frameworks [4, 5]. For training multilingual ASR systems, recent studies use transliterated text pooled across various languages [8, 9]. In the context of multilingual ASR systems for Indian languages, [10] used a simple character mapping based on CLS. The current work uses the phone-based CLS representation obtained using the rule-based unified parser [7] for training multilingual ASR systems.

This paper uses the CLS representation and proposes two different approaches to train multilingual E2E transformer [11] 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 (MT) 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, without using a separate LID and MT backend, we modify the E2E transformer framework to train the 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 MUCS 2021 challenge and the associated datasets are briefly described in Section 2. All the ASR systems proposed in this paper use the CLS transcription for training. The conversion of 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. MUCS 2021 Challenge

The MUCS 2021 challenge [2, 3] 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,

[*] equal contribution

Table 1: *Datasets provided in the MUCS 2021 challenge*

Language	Category	Duration (Hrs)	No. of unique sentences	Average no. of words per sentence
		Sub-Task 1		
Telugu (te)	Train	40	34,176	12.3
	Dev	5	2,997	9.4
Tamil (ta)	Train	40	30,239	7.3
	Dev	5	3,057	9.5
Gujarati (gu)	Train	40	20,208	12.3
	Dev	5	3,069	11.8
Hindi (hi)	Train	95.05	4,506	7.4
	Dev	5.55	386	12.2
Marathi (mr)	Train	93.89	2,543	6.8
	Dev	5	200	6.7
Odia (or)	Train	94.54	820	9.1
	Dev	5.49	65	9.1
		Sub-Task 2		
Hindi-English (hi-en)	Train	89.86	44244	12.0
	Dev	5.18	2893	12.0
Bengali-English (bn-en)	Train	46.11	22386	9.4
	Dev	7.01	3968	9.0

Gujarati, Tamil, Telugu, and Odia. It is to be noted that, in the blind test, the spoken utterances are provided without any language information. The ASR system has to identify the language and decode the given utterance in the language-specific script. Of the six languages provided for this task, only Hindi and Marathi share the *"Devanagari"* script, and the other four languages have their own script.

The goal of sub-task 2 is to build code-switching ASR systems. Hindi-English and Bengali-English are the two code-switched data pairs provided for this sub-task. Unlike sub-task 1, the language-pair information is provided in the blind set. However, the model should decode each word present in the code-switched utterance in the native script itself (Hindi/Bengali or English). The statistics of data provided for the two sub-tasks is presented in Table 1. For both the sub-tasks, the average WER computed across all languages/language pairs is used as the evaluation metric.

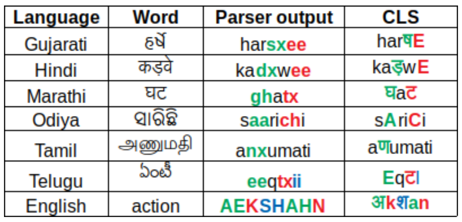
3. Common Label Set (CLS) and Unified Parser

The first step in the proposed approach is to convert the text in terms of the CLS format [6]. In CLS, similar phones across 13 Indian languages are mapped together and given a common label. Language-specific phones are given separate labels. The native text is converted to its constituent CLS representation using a unified parser [7]. 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 recognises the script based on the Unicode range. Then the parser applies the relevant language-specific rules and outputs the constituent phones in terms of CLS.

The CLS format uses a sequence of English characters for representation. Hence, to make the text purely phonetic, each CLS label is further mapped to a single character. For example, the CLS label "aa" (long vowel "a", as in *art*) is mapped to character "A", CLS label "ph" (aspirated stop consonant "p") to character "P", and so on. This CLS-like mapping was proposed in [4]. The word to final CLS mapping is illustrated by examples from different languages in Table 2.

In the code-switching task, English words are also present in the text. English words are first parsed through a neural network-based grapheme to phoneme converter [12], and the constituent phones are obtained in terms of the Carnegie Mellon University (CMU) phone set. These labels are then mapped to CLS 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.

Table 2: *Examples of words and their corresponding CLS representations*



Language	Word	Parser output	CLS
Gujarati	હર્ષે	harsxee	harषE
Hindi	कड़वे	kadxwee	kaड़wE
Marathi	घट	ghatx	घaट
Odiya	ସାରିଛି	saarichi	sAriCi
Tamil	அணுமதி	anxumati	aणumati
Telugu	ఏంటీ	eeqtxii	Eqटी
English	action	AEKSHAHN	अkशan

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.

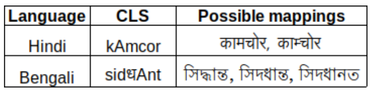
4.1. Baseline E2E Model

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 [7]. 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.

Table 3: *Confusions in CLS to native script mapping*



Language	CLS	Possible mappings
Hindi	kAmcor	कामचोर, काम्चोर
Bengali	sidधAnt	সিদ্ধান্ত, সিদধান্ত, সিদধানত

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 multinomial naive Bayes classifier using TF-IDF features obtained from decoded CLS is used to identify the language.

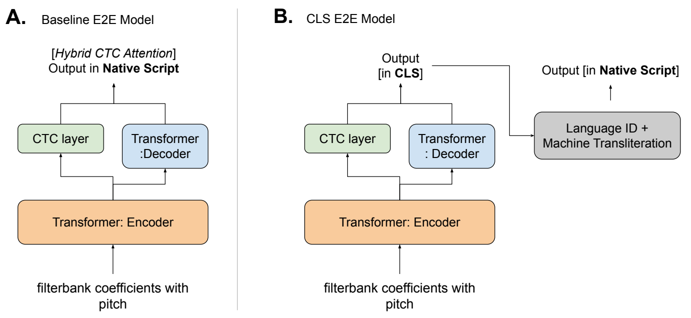
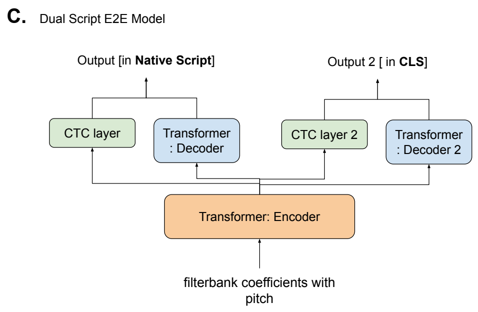

A. Baseline E2E Model
[Hybrid CTC Attention] Output in Native Script
CTC layer
Transformer :Decoder
Transformer: Encoder
filterbank coefficients with pitch
B. CLS E2E Model
Output [in CLS]
Output [in Native Script]
CTC layer
Transformer : Decoder
Language ID + Machine Transliteration
Transformer: Encoder
filterbank coefficients with pitch


Figure 1: *Illustration of proposed E2E hybrid CTC-attention models using CLS.* ***A****) baseline E2E model trained by pooling data for different languages.* ***B****) proposed E2E model trained on CLS representation with a separate backend for LID and MT.* ***C****) proposed dual script E2E model that uses both CLS and native script representation for training*

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.

Table 4: *Detailed transformer architecture used for E2E acoustic models*

Name	Value	Name	Value	Name	Value
Attention Dimensions	256	Encoder Layers	12	Decoder Layers	6
Attention Heads	4	Encoder Dimension	2048	Decoder Dimension	2048
Output BPU	5000	Output 2 BPU [1]	800		

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 challange (MUCS 2021) 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 multinomial Bayes classifier was used to predict the language. This system was also trained with IndicTTS text data [18] and ASR challange (MUCS 2021) 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].

[1]Used for output 2 (see Figure 1. C) in dual script model

5. Results

5.1. Results of sub-task 1

Table 5 shows our results on the development data for sub-task 1. Systems A and B (in Table 5) are provided by the challenge organizers [3]. The systems discussed in Section 4, without any language model, are represented as Systems C-H. Systems I-L denotes the same with a language model.

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 Systems J-L were submitted for evaluation on the blind test, and the results are given in Table 6. In contrast to results in Table 5, for the blind test, the challenge baseline 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.

Table 5: *Results of sub-task 1 on development data*

System ID	System Type	BPU/ CU	hi	mr	or	ta	te	gu	Avg
Baseline Methods (provided with MUCS 2021 Challenge [3])									
A	GMM-HMM	-	69.0	33.2	55.7	48.8	47.2	28.3	46.8
B	TDNN	-	40.4	22.4	39.0	33.5	30.6	19.2	30.7
Results without any Language Model									
C	Baseline	BPU	52.1	33.8	71.3	31.3	32.9	26.5	49.5
D	E2E Model	CU	26.5	**17.1**	**36.1**	35.3	36.6	28.4	30.0
E	CLS	BPU	34	21.8	50.1	31.7	31.5	26.5	32.6
F	E2E Model	CU	26.2	17.4	39.5	37.8	37.2	30.1	34.6
G	Dual Script	BPU	29.4	19.8	44.9	**30.5**	**31.9**	**24.4**	30.1
H	E2E Model	CU	**25.9**	**17.1**	37.4	35.2	35.8	27.7	**29.8**
Results with Language Model									
I	CLS	BPU	31.8	21.8	48.2	25.6	24.2	20.7	28.7
J	E2E Model	CU	**21.4**	**14.6**	38.3	28.8	27.3	22.4	25.4
K	Dual Script	BPU	27.8	20.0	48.2	**23.6**	**23.6**	**18.8**	27.0
L	E2E Model	CU	21.6	**15.1**	**36.0**	25.9	25.3	20.5	**24.0**

Table 6: *Results of sub-task 1 on blind data*

System ID	hi	mr	or	ta	te	gu	Avg-1	Avg-2
Baseline								
B	37.2	**29.0**	38.4	34.0	31.4	26.1	**32.73**	33.4
Final Systems Submitted								
J	19.5	85.9	37.1	32.0	30.3	32.9	39.6	30.3
K	25.3	100.3	51.2	**25.1**	**25.4**	**25.4**	42.1	30.4
L	**17.8**	111.7	**32.1**	27.1	28.1	29.8	41.1	**27.1**

Table 7: *Results of sub-task 2 on development and blind data*

System	System	BPU/	Dev Data			Blind Test		
ID	Type	CU	hi-en	bn-en	Avg	hi-en	bn-en	Avg
Baseline Methods (provided with MUCS 2021 Challenge [3])								
M	GMM-HMM	-	44.3	39.1	41.7	-	-	-
N	TDNN	-	36.9	34.0	35.6	-	-	-
O	E2E Model	BPU	27.7	37.2	32.4	25.5	32.8	29.1
Results without any Language Model								
P	Dual Script	CU	33.0	27.0	30	-	-	-
Q	E2E Model	BPU	28.9	**25.3**	27.1	**22.0**	27.8	24.9

5.2. Results of sub-task 2

Table 7 shows the baseline and proposed results on sub-task 2 development and blind data. Systems M-O represents the baseline results obtained by the challenge organizers (in [3]) and Systems P and Q denote the proposed models.

Unlike results in Section 5.1, the byte-pair models are observed to give better results than the character-level models. The dual script system with byte-pairs (System Q) gave $\approx 5\%$ improvement in WER over the baseline system (System O) on both development and blind data sets. Also, we highlight that this system was trained with combined Hindi-English and Bengali-English data, and the E2E model predicts the language pair during decoding (similar to sub-task 1). In contrast, the baseline system was trained separately for each language pair and decoded with language-pair information.

6. Discussion

6.1. CLS for E2E ASR system

In this paper, we propose two novel multilingual ASR systems that are trained with CLS. From the results in Table 5, 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 adds evidence to our conjecture that CLS cues the E2E model in learning common sounds across languages in a low resource setting.

The best system among the proposed CLS E2E and the dual script can not be determined from the results in Table 5. Note that for the CLS E2E model, LID and MT back-end performance were not compared against other state-of-art approaches. It can be an interesting future work to benchmark and improve the performance of the CLS E2E model.

In Table 6, we observe poor results for Marathi consistently. We note that the possible reason for this poor performance is the different channel encoding scheme between the Marathi blind and non-blind (train & test) datasets [3].

The dual script model provided a straightforward extension to the code-switching task (sub-task 2) due to the end-to-end training. For this task (in Table 7), the dual script model was able to improve the baseline results by 5% WER without using any language model.

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 [2].

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” (CS2021012MEIT003119) and “Speech to Speech Machine Translation” (CS2021152OPSA003119), funded by the Ministry of Electronics and Information Technology (MeitY) 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. Rajpuria, C. Yarra, A. 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," *arXiv preprint arXiv:2104.00235*, 2021.

[4] A. Prakash, A. Leela Thomas, S. Umesh, and H. A Murthy, "Building Multilingual End-to-End Speech Synthesisers 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 Synthesisers 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 Shanmugam, R. Krishnan, S. Kishore, K. Samudravijaya, 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. Emond, 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, "g2pe," 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 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. [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 ctc/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. Kudo, "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. Bhattacharyya, 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.