Investigating Transformer architecture for Automatic Speech Recognition and Machine Translation in Indian Languages

A Project Report

Submitted by

MOLUGU SAI SANTOSH GAURAV

in partial fulfillment of the requirements for the award of the degree

of

BACHELOR OF TECHNOLOGY & MASTER OF TECHNOLOGY

DEPARTMENT OF ELECTRICAL ENGINEERING
INDIAN INSTITUTE OF TECHNOLOGY MADRAS
CHENNAI-600036

JUNE 2021

THESIS CERTIFICATE

This is to certify that the thesis entitled **"Investigation of Transformer architecture for Automatic Speech Recognition and Machine Translation in Indian Languages"** submitted by **Molugu Sai Santosh Gaurav** to the Indian Institute of Technology, Madras for the award of the degree of **Bachelor of Technology and Master of Technology** is a bona fide record of research work carried out by him under my supervision. The contents of this thesis, in full or in parts, have not been submitted to any other Institute or University for the award of any degree or diploma.

Prof. Umesh S.
Associate Professor
Department of Metallurgical and Materials Engineering
Indian Institute of Technology Madras
Chennai – 600 036.

Place: Chennai

Date: 3rd June 2021

ACKNOWLEDGEMENTS

I am grateful to my research guide Professor Umesh S for his constant invaluable guidance and support throughout the project. Also, special thanks to Aditya, Vishwas and Vrunda for resolving my doubts and helping me in my thought process. This project has helped me learn a few crucial lessons from this project regarding a first hand experience of research and formalised my thought process in conveying my ideas.

ABSTRACT

Keywords: Transformer architecture, Neural Machine Translation (NMT), Automatic Speech Recognition (ASR), Common Label Set (CLS), Indian Languages.

The core of the present work deals with exploring the use of transformer architecture in the areas of NMT and ASR modelling for Indian Languages.

Transformer encoder-decoder models have become popular in natural language processing [2] and speech processing [3]. The Transformer architecture allows to successfully train a deep stack of self-attention layers via residual connections [21] and layer normalisation [22]. The positional encodings [2], typically based on sinusoidal functions, are used to provide the self-attention with the sequence order information.

Across various applications, systematic improvements have been reported over the standard, multi-layer long short-term memory (LSTM) recurrent neural network based models. Transformer based End-To-End NMT models have shown to be effective in modelling text data, the transformer architecture incorporates self-attention mechanisms to effectively model the spatial correlations found in text. In the case of machine translation transformers have outperformed the previous benchmarks set by GoogleNMT.

Transformer based End-To-End Automatic Speech Recognition (ASR) systems work very well in a multilingual scenario due to their power of encapsulating the acoustic, pronunciation, and language models in a single network. Common label representation has proven to be very effective in building ASR systems for low resource Indian languages. Generating a Common Label Set (CLS) [5] for these languages is easy due to the similar ordering of characters based on the sounds they represent.

In this report we explore the use of Transformers in modelling Machine Translation for Indian Languages with emphasis on academic lecture data and multilingual speech recognition for six Indian Languages.

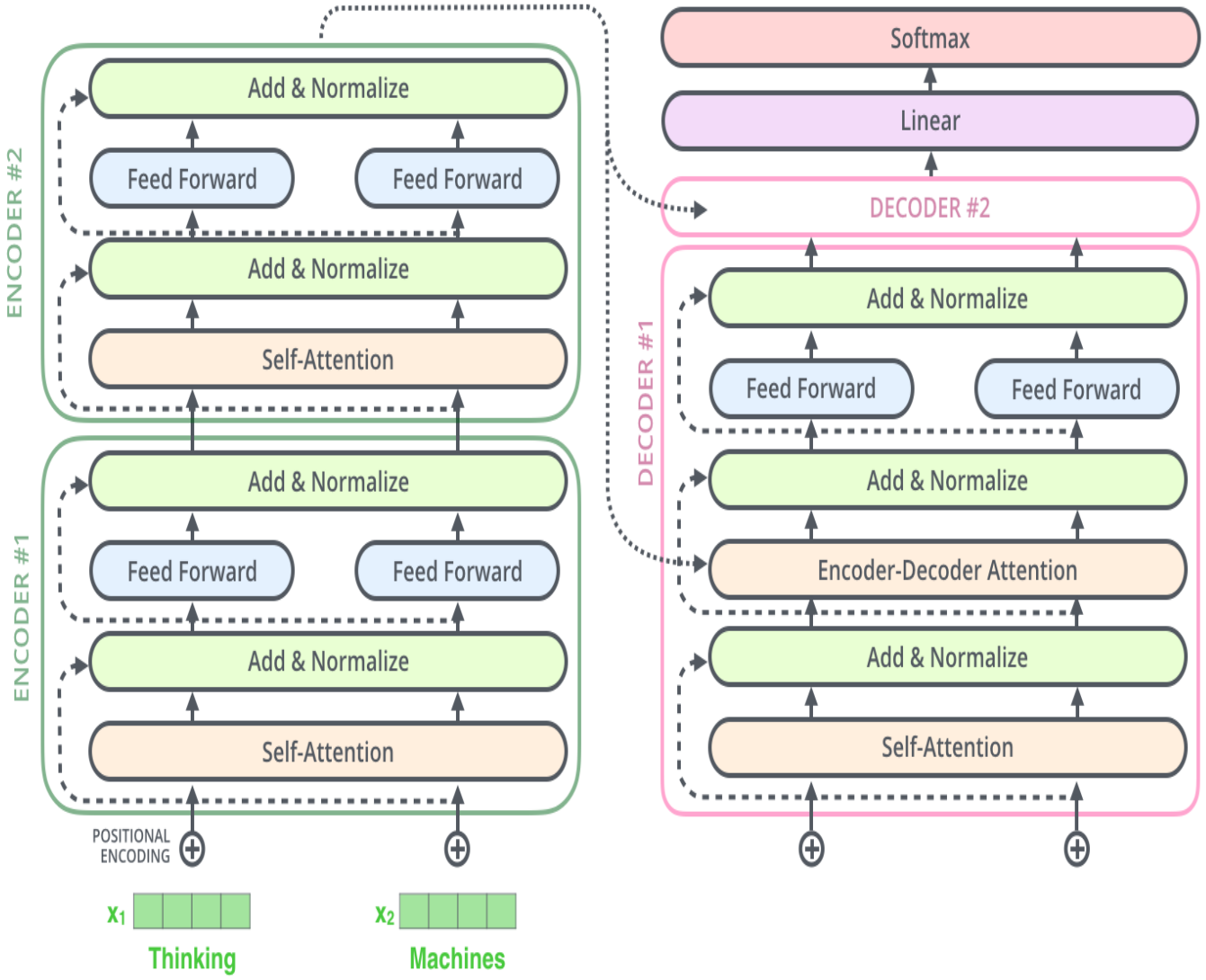

Fig. i Complete visualisation of an End-To-End Transformer model. (Courtesy: Alammar, J (2018).The Illustrated Transformer [Blog post]. Retrieved from https://jalammar.github.io/illustrated-transformer/). ii

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1
TRANSFORMERS IN NEURAL MACHINE TRANSLATION FOR INDIAN LANGUAGES

1.1. Introduction

NMT refers to the use of computers to automate some of the tasks or the entire task of translating between human languages. Given a sequence of text in a source language, there is no one single best translation of that text to another language. This is because of the natural ambiguity and flexibility of human language. This makes the challenge of automatic machine translation difficult, perhaps one of the most difficult in artificial intelligence. Classical machine translation methods often involve rules for converting text in the source language to the target language. The rules are often developed by linguists and may operate at the lexical, syntactic, or semantic level. This focus on rules gives the name to this area of study: Rule-based Machine Translation, or RBMT.

Neural machine translation, or NMT for short, is the use of neural network models to learn a statistical model for machine translation. The key benefit to the approach is that a single system can be trained directly on source and target text, no longer requiring the pipeline of specialized systems used in statistical machine learning. As such, neural machine translation systems are said to be end-to-end systems as only one model is required for the translation.

Multilayer Perceptron neural network models can be used for machine translation, although the models are limited by a fixed-length input sequence where the output must be the same length. These early models have been greatly improved upon recently through the use of recurrent neural networks organized into an encoder-decoder architecture that allow for variable length input and output sequences. Key to the encoder-decoder architecture is the ability of the model to encode the source text into an internal fixed-length representation called the context vector. Interestingly, once encoded, different decoding systems could be used, in principle, to translate the context into different languages. Although effective, the Encoder-Decoder architecture has problems with long sequences of text to be translated. The problem stems from the fixed-length internal representation that must be used to decode each word in the output sequence. The solution is the use of an attention mechanism that allows the model to learn where to place attention on the input sequence as each word of the output sequence is decoded.

The encoder-decoder recurrent neural network architecture with attention is currently the state-of-the-art on some benchmark problems for machine translation. And this architecture is used in the heart of the Google Neural Machine Translation system, or GNMT, used in their Google Translate service.

Development of a full-fledged bilingual MT system for any two natural languages with limited electronic resources and tools is a challenging and demanding task. The current work in NMT for Indian Languages has mainly focused on general purpose English to Indian Language translations. However, these models fail in translation of technical academic data. The end-to-end NMT systems require a huge amount of training data to train, many Indian languages do not have large english transcribed data to build good English to Indian Language NMT modelling. In the case of technical data, the data is even scarce, this kind of modelling is referred to as low resource NMT modelling.

Apart from major cities and towns in India, the majority of the students are given education in their non-English mother tongue. To make education more accessible to students across India we have explored the idea of building a NMT model for Indian language for technical academic data. With the help of these NMT models one can easily access the course information in their native language.

1.2. Dataset Details

There are 4 different datasets used in the NMT experiments, a 1.4 M general purpose en-hi corpus curated by IITB (General purpose en-hi), a ~300 K technical en-hi corpus curated from NPTEL lectures using nltk and regex libraries by IITM Speech Lab (Technical en-hi), a 8.56 M en-hi corpus curated by AI4Bharat (IITB data is included), and a small dataset of 10K corpus curated from SRT text.

1.3. Experimental Details and Discussion

1.3.1. Parameters

The modelling of English to Hindi Neural Machine Translation are carried out on ESPnet1 [20] transformer framework with 6 Encoder, 6 Decoder, 2048 Linear units, 4 head attention heads, 256 attention dimension, bpe encoding transformer model. The experiments in Table

1.3.2.1, 1.3.2.2, 1.3.2.3 were done on 1 NVIDIA GTX 1080 Ti card with batch size of 32 and accumgrad of 2, the experiments in Table 1.3.2.4 were done on 1 NVIDIA A100 card with batch size of 32 and accumgrad of 2.

1.3.2. Results

The BLEU scores, Brevity Penalty and Ratio metrics for both the general purpose and technical en-hi NMT models are provided in Table 1.3.2.1, 1.3.2.2, 1.3.2.3, 1.3.2.4.

Train Dataset	Data size	nbpe	Pre-train Model	Test Dataset	BLEU score	BP	Ratio
IITB	1.4 M	16 K	None	IITB	22.67	0.911	0.915
IITB	1.4 M	10 K	None	IITB	24.81	0.928	0.931
IITB	1.4 M	7.5 K	None	IITB	26.54	1.000	1.005
IITB	1.4 M	5 K	None	IITB	26.02	1.000	1.001

Table 1.3.2.1: Performance with respect to number of bpe (byte pair encoding).

The number of byte pair encodings (nbpe) is an important hyperparameter in modelling a NMT model, the nbpe is dependent on the vocabulary size or training data size. In Table 1.3.2.1, we find that the 7.5 K nbpe gives the best performance for 1.4 M training data.

Train Dataset	Data size	nbpe	Pre-train Model	Test Dataset	BLEU score	BP	Ratio
IITB + NPTEL	1.4 M + 45 K	10 K	None	IITB	27.11	1.000	1.260
IITB + NPTEL	1.4 M + 45 K	10 K	None	NPTEL	21.33	1.000	1.239

Table 1.3.2.2: Performance with respect to addition of small amounts of technical data.

To assess the impact of addition of technical data(NPTEL) to General Purpose data(IITB), a small amount of technical data from 300 K NPTEL corpus is added and a transformer is trained. As shown in Table 1.3.2.2, the addition of data has shown to significantly improve the General en-hi performance and help us arrive at a baseline regarding the technical en-hi performance.

Model ID	Train Dataset	Finetune Dataset	Data size	nbpe	Pre-train Model	Finetune epochs	Test Dataset	BLEU score	BP	Ratio
Model1	IITB + NPTEL	None	1.4 M + 300 K	7.5 K	None	None	IITB	21.27	0.793	0.812
Model1	IITB + NPTEL	None	1.4 M + 300 K	7.5 K	None	None	NPTEL	17.61	0.740	0.769
Model1	IITB + NPTEL	None	1.4 M + 300 K	7.5 K	None	None	SRT	33.60	0.801	0.818
Model2	IITB + NPTEL	NPTEL	1.4 M + 300 K	7.5 K	Model1	5	IITB	21.40	0.790	0.810
Model2	IITB + NPTEL	NPTEL	1.4 M + 300 K	7.5 K	Model1	5	NPTEL	18.06	0.738	0.767
Model2	IITB + NPTEL	NPTEL	1.4 M + 300 K	7.5 K	Model1	5	SRT	34.13	0.808	0.824

Table 1.3.2.3: Comparing the pre training and fine tuning performance of en-hi transformer NMT.

Fine Tuning is a method employed to enhance the performance of a model for a particular task. To improve the low resource NPTEL performance, a model trained on General purpose and Technical corpora is trained and later fine tuned for a few epochs. The Table 1.3.2.3 shows the increase in performance when Model2 is finetuned towards Technical en-hi NMT using Model1 for 5 epochs.

Model ID	Train Dataset	Data size	nbpe	Pre-train Model	Finetune epochs	Test Dataset	BLEU score	BP	Ratio
Model 3	SAMANANTAR + NPTEL	8.4 M + 300 K	10 K	None	None	IITB	28.46	0.816	0.831
Model 3	SAMANANTAR + NPTEL	8.4 M + 300 K	10 K	None	None	NPTEL	27.36	0.805	0.822
Model 4	SAMANANTAR + NPTEL	8.4 M + 300 K	10 K	Model 3	5	NPTEL	32.40	0.799	0.817

Table 1.3.2.4: Final NMT for General Purpose(Model3) and Technical(Model4) en-hi translation.

Finally, we train a NMT model using all the data available and later finetune the model using technical corpus, a higher nbpe is used to accommodate the increase in vocabulary or dataset

size. Table 1.3.2.4 shows the best performance achieved on General purpose as well as Technical en-hi translation, the table also shows that with increase in data the models perform better.

1.4. Conclusion and Future Work

From the above experiments we see that transformers work reasonably well even with limited amounts of data. With more data, the model improves in performance significantly. As data for Indian languages is currently scarce, we could use less complex transformer models to train NMT models for other Indian languages. The decoder part of the transformer in a way acts as a language model and this helps in building faster and less complex NMT models without the need for language models.

The performance could be further improved by addition of new data, using back translation [23], exploring the integration of BERT and GPT models. Recently, conformers [8] have found to be outperforming transformers as it uses localised attention in modelling.

CHAPTER 2

TRANSFORMER IN MULTILINGUAL AUTOMATIC SPEECH RECOGNITION FOR INDIAN LANGUAGES

2.1. INTRODUCTION

End-To-End ASR models obviate the need for lexicon and language models and offer better performance than conventional hybrid systems. Also these models can generalize better to unseen words, which makes them suitable for low resource multilingual speech recognition. With the advent of attention [1] between the encoder and decoder, performance of the sequence-to-sequence models have further improved. This includes the new Transformer [2] architecture, solely based on attention. Though Transformers were introduced for Natural Language Processing (NLP), they are gaining popularity in ASR too [3, 4, 5, 6]. Conformer [7], a hybrid of Transformer and Convolutional Neural Network, which is more suited for speech, is also gaining popularity in recent times.

A major problem in building good ASR systems for many languages is the lack of transcribed data. In order to tackle the data insufficiency problem, most works have focused on pooling data from different languages to build multilingual systems. Multilingual ASR systems trained with data from many languages, not only have an inherent advantage of recognising speech corresponding to different languages used on training but also solve the data insufficiency problem by pooling. [8] explored a common acoustic model for multiple languages based on a global phone set. Later, [9] explored different training approaches using tandem and bottleneck features for multilingual Multi-layer perceptron models. In [10], a hybrid attention-CTC model was employed to perform grapheme-based speech recognition of 10 different languages. A similar idea of using a single multilingual end-to-end model based on attention was shown to improve performance over monolingual models for various Indian languages with the grapheme target being a union of all the characters in the languages considered[11]. The application of adapter modules was explored for the purpose of handling imbalanced datasets in a multilingual scenario in [12]. In multilingual systems, as the number of languages increases, the number of targets to be modelled also increases.

The concept of using a common phone set, which reduces the target size, was earlier used in English-Hindi code switched ASR [13, 14] and also in multilingual speech synthesizers across four Indian languages [15]. In our previous work in multilingual ASR [5], we proposed a Common Labeled set (CLS) to map characters with similar sounds from different Indian languages to one common representation due to similar characteristics across Indian languages. We used CLS in a Transformer framework and showed an improvement in performance compared to plain multilingual models, as the number of characters required to be modeled by the network is less in CLS. Though CLS gives improvement, to render the decodes in their original language, prior language information is necessary.

In this paper, we explore CLS further in multilingual speech recognition models for six low resource Indian languages: Hindi, Marathi, Odia, Tamil, Telugu, and Gujarati in a CLS setup. We investigate the following: (i) performance comparison of multilingual CLS models in transformer framework for a new set of Indian languages (ii) performance comparison of different languages when the data is split on the basis of language family and vocabulary size.

2.2 DATASET DETAILS

The datasets used in the experiments are a part of the "Multilingual and code-switching ASR challenges for low resource Indian languages, Interspeech 2021". The dataset comprises audio and corresponding text from 6 different Indian Languages: Hindi, Marathi, Odia, Tamil, Telugu, and Gujarati. Microsoft originally released the Tamil, Telugu, and Gujarati datasets as a part of "Low resource Speech Recognition Challenge On Indian Languages, Interspeech 2018". The Hindi and Marathi datasets are collected from a collection of stories, whereas the Odia is based on Agriculture, Healthcare, and Finance domains, and the Tamil, Telugu, and Gujarati data are from a general domain. Hindi, Marathi and Odia have about 95 hours of train data each and the sampling rate is 8 kHz. Tamil, Telugu and Gujarati have about 40 hours each of train data and the sampling rate is 16 kHz.

Language	Encoding	Duration(hrs)	Vocabulary Size
Hindi	16bit,8kHz	Train - 95.05	6542

		Test - 5.55	
Marathi	16bit,8kHz	Train - 93.89 Test - 5	1644
Odia	16bit,8kHz	Train - 94.54 Test - 5.49	3395
Tamil	16bit,16kHz	Train - 40 Test - 5	50123
Telugu	16bit,16kHz	Train - 40 Test - 5	43720
Gujarati	16bit,16kHz	Train - 40 Test - 5	39327

Table 2.2.1: Dataset Details.

2.3 Common Label Set

A unique characteristic of Indian family of languages is that there is a uniform order in which the alphabets are placed across most of the languages. Also, there is a strong correspondence between alphabets and the phonemes they represent. This is shown in Table 2.3.1. For example, the first unicode character in all indian languages represents the phoneme /a/. In our previous work [5], we explored the benefits of pooling similar sounding characters across languages and representing them as one single unit, i.e., by a Common Label Set (CLS). The most interesting part here is the ease with which the CLS can be generated for Indian Languages. Due to the logical ordering of characters just by knowing the position of a character in the alphabetical list of any Indian language, we can easily map it to the character and sound that it represents in the new language. Our experiments in [5] showed that a CLS based multilingual model is better than the regular multilingual model based on the union of all characters.

Hindi	अ	आ	-	-	इ	ई	उ	ऊ	ऋ	-	ए	ऐ	-	ओ	-	औ	क	ख	ग	घ	ङ	-	च	छ	ज	झ	ञ	-	ट	ठ	ड	ढ	ण	-	त	थ	द	ध	न	-	प	फ	ब	भ	म
Marathi	अ	आ	-	-	इ	ई	उ	ऊ	ऋ	-	ए	ऐ	-	ओ	-	औ	क	ख	ग	घ	ङ	-	च	छ	ज	झ	ञ	-	ट	ठ	ड	ढ	ण	-	त	थ	द	ध	न	-	प	फ	ब	भ	म
Odia	ଅ	ଆ	-	-	ଇ	ଈ	ଉ	ଊ	ଋ	-	ଏ	-	-	ଓ	-	ଔ	କ	ଖ	ଗ	ଘ	ଙ	-	ଚ	ଛ	ଜ	ଝ	ଞ	-	ଟ	ଠ	ଡ	ଢ	ଣ	-	ତ	ଥ	ଦ	ଧ	ନ	-	ପ	ଫ	ବ	ଭ	ମ
Gujarati	અ	આ	-	-	ઇ	ઈ	ઉ	ઊ	ઋ	ઍ	એ	ઐ	ઑ	ઓ	-	ઔ	ક	ખ	ગ	ઘ	ઙ	-	ચ	છ	જ	ઝ	ઞ	-	ટ	ઠ	ડ	ઢ	ણ	-	ત	થ	દ	ધ	ન	-	પ	ફ	બ	ભ	મ
Telugu	అ	ఆ	-	-	ఇ	ఈ	ఉ	ఊ	ఋ	ఎ	ఏ	ఐ	-	ఒ	ఓ	ఔ	క	ఖ	గ	ఘ	ఙ	-	చ	ఛ	జ	ఝ	ఞ	-	ట	ఠ	డ	ఢ	ణ	-	త	థ	ద	ధ	న	-	ప	ఫ	బ	భ	మ
Tamil	அ	ஆ	-	-	இ	ஈ	உ	ஊ	-	எ	ஏ	ஐ	-	ஒ	ஓ	ஔ	க	-	-	-	ங	-	ச	-	ஜ	-	ஞ	-	ட	-	-	-	ண	-	த	-	-	-	ந,ன	-	ப	-	-	-	ம
Phoneme	a	aa	æ	æ̃	i	ii	u	uu	rq	e	ee	ai	ax	o	oo	au	k	kh	g	gh	ng	ṅga	c	ch	j	jh	nj	gna	tx	txh	dx	dxh	nx	ṅḍa	t	th	d	dh	n	ṅda	p	ph	b	bh	m

Table 2.3.1: Character to Common Phoneme Map

2.4. EXPERIMENTAL DETAILS AND DISCUSSION

All 8 kHz data has been upsampled to 16 kHz. In the case of multilingual models, we take the union of BPEs in the native script as targets. In this section we explore the Multilingual and Multilingual+CLS in Transformer framework.

2.4.1. TRANSFORMERS

Transformers are sequence to sequence models which are very popular in Natural Language Processing (NLP). The performance improvement in Transformers come from the self attention modules which allows them to model global context. Transformers have also found their way into ASR and give good results [3, 6]. For ASR, rather than utterance level attention, local patterns at phrase level are more important but vanilla Transformers are not able to exploit them.

2.4.2. TRANSFORMER EXPERIMENTS

2.4.2.1. TRANSFORMER PARAMETERS

For the 6 language multilingual ASR model and 6 language CLS model, the number of encoders, decoders are 12 and 6 respectively. The attention dimension of the corresponding model is 384 and the number of attention heads is 6. The number of encoder and decoder units is 2048 and 1000 byte pair encodings were used, in order to account for the character sets of all the 6 languages. For all the experiments, specaug [19] was employed with two masks each for frequency and time. Range of 40 was used for frequency mask and 30 was used for time mask. The training was done on 1 NVIDIA GTX 1080 Ti card with a batch size of 32 and accumgrad of 2. The models mentioned above were trained on a linear combination of attention loss and ctc loss with the ctc loss parameter being 0.3 using espnet toolkit [20].

2.4.2.2. TRANSFORMER RESULTS

Table 2.4.2.2.1 presents the WERs for Multilingual and Multilingual+CLS. From the results we see that the performance of Multilingual+CLS is consistently better than Multilingual for all the six languages.

Language	Multi	Multi+CLS
Hindi	35.8	35.2
Marathi	21.9	21.7
Odia	50.2	48.5
Tamil	31.9	27.9
Telugu	32.9	28.2
Gujarati	26.9	22.1

Table 2.4.2.2.1: WER on test set for transformer architecture on multilingual ASR and CLS model for each of the 6 languages.

2.4.3. ANALYSING THE PERFORMANCE BY PARTITIONING THE LANGUAGES

We partitioned the languages on the basis of vocabulary size and it is named as SPLIT 1 in 6. The vocabulary sizes for Hindi, Marathi, Odia, Tamil, Telugu, and Gujarati are 6542, 1644, 3395, 50123, 43720, 39327, respectively. Hereafter, the combined set of Hindi, Marathi, Odia languages for SPLIT 1 will be referred to as SET1 correspond to low vocabulary languages, and combined set of Tamil, Telugu, Gujarati will be referred to as SET2 correspond to large vocabulary. Despite the duration of the train sets of languages in SET1 being significantly higher than those in SET2, the vocabulary is smaller since SET1 was collected by making different speakers read the same set of sentences. Thus, we noticed a lot of repeated sentences.

Two major families of Indian languages exist: Indo-Aryan, spoken by the people predominantly in North and Central India, and Dravidian, spoken in Southern India. Hence,

SPLIT 2 as seen in Table 5.3.1.1 is the partition based on language family. In this partitioning, Tamil and Telugu were considered as one group as they belong to the Dravidian family of languages and remaining languages formed the other group as they belong to Indo-Aryan family of languages.

2.4.3.1 RESULTS

The experiments were performed in the transformer framework and the parameters were retained the same as multilingual+cls.
From the Table 2.4.3.1.1, these are the main inferences:
1. The WERs of Hindi, Marathi, and Odia, in both SPLIT 1 and SPLIT 2 have improved. This might be due to the pooling of small vocabulary size of these languages, although these are 90 hour duration datasets.
2. Whenever languages with rich vocabulary are added, the ones with less vocabulary benefit significantly.
3. In SPLIT 2 dravidian languages Tamil and Telugu degrades over the 6-language baseline as the data size is 2/3 times the SET1.
4. SPLIT 2 seems to help Indo-Aryan languages. This might be owing to the fact that they are getting help from the one large vocabulary datasets (Gujarati) reinforcing the second point.

Language	**6-lang (Multi + CLS)**	**SPLIT 1**	**SPLIT 2**
Hindi	35.2	34.8	31.5
Marathi	21.7	20.3	19.9
Odia	48.5	40.6	43.2
Tamil	27.9	24.4	29.2
Telugu	28.2	25.8	30.9
Gujarati	22.1	20.7	27.1

Table 2.4.3.1.1: This table compares the performance of the different languages when we make divisions of the dataset based on language family and data characteristics.

2.5. CONCLUSION

Common Label Set performs better even for a new set of languages. The performance degradation over the multilingual+CLS is not very significant. We also analysed the datasets by partitioning them on the basis of vocabulary size and language family resulting in some interesting insights: i) For the CLS to benefit, vocabulary size of the datasets pooled must be high. ii) Grouping the languages based on language families helps improve the performance. We would like to explore self-supervised pre-training methods for multilingual models.

REFERENCES

[1] D. Bahdanau, K. Cho, and Y. Bengio, “Neural machine translation by jointly learning to align and translate,” arXiv preprint arXiv:1409.0473, 2014.

[2] 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.

[3] L. Dong, S. Xu, and B. Xu, “Speech-transformer: A no-recurrence sequence-to-sequence model for speech recognition,” in 2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 2018, pp. 5884–5888.

[4] 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.

[5] V. M. Shetty, M. Sagaya Mary N J, and S. Umesh, “Exploring the use of common label set to improve speech recognition of low resource indian languages,” in ICASSP 2021 - 2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 2021.

[6] M. S. Mary N J, V. M. Shetty, and S. Umesh, “Investigation of methods to improve the recognition performance of tamil-english code-switched data in transformer framework,” in ICASSP 2020- 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 2020, pp. 7889–7893.

[7] 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.

[8] T. Schultz and A. Waibel, “Fast bootstrapping of lvcsr systems with multilingual phoneme sets,” in Fifth European Conference on Speech Communication and Technology, 1997.

[9] S. Thomas, S. Ganapathy, and H. Hermansky, “Multilingual mlp features for low-resource lvcsr systems,” in 2012 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). IEEE, 2012, pp. 4269–4272.

[10] S. Watanabe, T. Hori, and J. R. Hershey, “Language indepen- dent end-to-end architecture for joint language identification and speech recognition,” in 2017 IEEE

Automatic Speech Recognition and Understanding Workshop (ASRU). IEEE, 2017, pp. 265–271.

[11] S. Toshniwal, T. N. Sainath, R. J. Weiss, B. Li, P. Moreno, E. Weinstein, and K. Rao, "Multilingual speech recognition with a single end-to-end model," in 2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 2018, pp.4904–4908.

[12] A. Kannan, A. Datta, T. N. Sainath, E. Weinstein, B. Ramabhad-ran, Y. Wu, A. Bapna, Z. Chen, and S. Lee, "Large-scale multi-lingual speech recognition with a streaming end-to-end model," arXiv preprint arXiv:1909.05330, 2019.

[13] S. Sivasankaran, B. M. L. Srivastava, S. Sitaram, K. Bali, and M. Choudhury, "Phone merging for code-switched speech recognition," in Third Workshop on Computational Approaches to Linguistic Code-switching, 2018.

[14] K. Dhawan, G. Sreeram, K. Priyadarshi, and R. Sinha, "Investigating target set reduction for end-to-end speech recognition of hindi-english code-switching data," in 2020 National Conference on Communications (NCC). IEEE, 2020, pp. 1–5.

[15] A. Prakash, A. L. Thomas, S. Umesh, and H. A. Murthy, "Building multilingual end-to-end speech synthesisers for indian languages," in Proc. of 10th ISCA Speech Synthesis Workshop (SSW'10), 2019, pp. 194–199.

[16] D.-C. Lyu and R.-Y. Lyu, "Language identification on code-switching utterances using multiple cues," in Ninth Annual Conference of the International Speech Communication Association, 2008.

[17] K. R. Mabokela and M. J. Manamela, "An integrated language identification for code-switched speech using decoded-phonemes and support vector machine," in 2013 7th Conference on Speech Technology and Human-Computer Dialogue (SpeD). IEEE, 2013, pp. 1–6.

[18] J. Gonzalez-Dominguez, D. Eustis, I. Lopez-Moreno, A. Senior, F. Beaufays, and P. J. Moreno, "A real-time end-to-end multilin-gual speech recognition architecture," IEEE Journal of selected topics in signal processing, vol. 9, no. 4, pp. 749–759, 2014.

[19] D. S. Park, W. Chan, Y. Zhang, C.-C. Chiu, B. Zoph, E. D. Cubuk, and Q. V. Le, "SpecAugment: A Simple Data Augmentation Method for Automatic Speech Recognition,"

in Proc. Interspeech 2019, 2019, pp. 2613–2617. [Online]. Available: http://dx.doi.org/10.21437/Interspeech.2019-2680

[20] S. Watanabe, T. Hori, S. Karita, T. Hayashi, J. Nishi-toba, 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

[21] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in IEEE Conf. on Computer Vision and Patt. Recog. (CVPR), Las Vegas, NV, USA, June. 2016, pp. 770–778.

[22] J. L. Ba, J. R. Kiros, and G. E. Hinton, "Layer normalization," arXiv preprint arXiv:1607.06450, 2016.

[23] Rico Sennrich, Barry Haddow, Alexandra Birch, "Improving Neural Machine Translation Models with Monolingual Data" ,CoRR ,2015.