IndicBART: A Pre-trained Model for Natural Language Generation of Indic Languages

Raj Dabre\(^1\) Himani Shrotriya\(^2\) Anoop Kunchukuttan\(^4\)
Ratish Puduppully\(^4\) Mitesh M. Khapra\(^5\) Pratyush Kumar\(^6\)

National Institute of Information and Communications Technology\(^1\) IIT Madras\(^2,5,6\)
Microsoft\(^3,4\) University Of Edinburgh\(^4\)
\(^1\)raj.dabre@nict.go.jp \(^2\)cs20m024@smail.iitm.ac.in
\(^3\)ankunchu@microsoft.com \(^4\)r.puduppully@sms.ed.ac.uk
\(^5\)miteshk@cse.iitm.ac.in \(^6\)pratykumar@microsoft.com

Abstract

In this paper we present IndicBART, a multilingual, sequence-to-sequence pre-trained model focusing on 11 Indic languages and English. Different from existing pre-trained models, IndicBART utilizes the orthographic similarity between Indic scripts to improve transfer learning between similar Indic languages. We evaluate IndicBART on two NLG tasks: Neural Machine Translation (NMT) and extreme summarization. Our experiments on NMT for 12 language pairs and extreme summarization for 7 languages using multilingual fine-tuning show that IndicBART is competitive with or better than mBART50 despite containing significantly fewer parameters. Our analyses focus on identifying the impact of script unification (to Devanagari), corpora size as well as multilingualism on the final performance. The IndicBART model is available under the MIT license at https://indicnlp.ai4bharat.org/indic-bart.

1 Introduction

With the adoption of deep learning models in natural language processing, there has been significant progress in natural language generation for various tasks like machine translation, abstractive summarization, data-to-text generation etc. (Wu et al., 2016; Paulus et al., 2018; Puduppully et al., 2019). The NLG models are typically different types of attention-based sequence-to-sequence models (conditional language models), which have been successful in generation of fluent and relevant output on various tasks. In line with the trend toward unsupervised pre-training, pre-trained sequence-to-sequence models have been shown to be useful to improve performance on various NLG tasks (Rothe et al., 2020; Kale and Rastogi, 2020; Lewis et al., 2020). The pre-trained models enable the network to implicitly learn about language structure, language model and external knowledge. The pre-trained models can then be finetuned for downstream tasks and are particularly useful in low-resource scenarios. Multilingual pre-trained models trained on many languages and multilingual finetuning methods are also widely used (Tang et al., 2020; Liu et al., 2020; Xue et al., 2021).

Following this line of research, in this paper, we present IndicBART, a multilingual pre-trained sequence to sequence model specifically trained for Indic languages, which are spoken by more than a billion users. While universal pre-trained models like mT5 (Xue et al., 2021), mBART25 (Liu et al., 2020) and mBART50 (Tang et al., 2020) support a large number of languages, we believe there is a case for building a dedicated model for Indian languages. By restricting our focus to Indic languages alone, we can use the model capacity more optimally and build smaller models for Indic languages that are competitive with the above mentioned universal models (Liu et al. (2020) suggest that increasing number of languages degrades performance when monolingual data is plentiful). These models can be finetuned on comparatively modest hardware for downstream tasks. Previous work has shown the benefits of pre-trained language models as well as NMT models that cater to a set of related languages (Kakwani et al., 2020; Tan et al., 2019; Khanuja et al., 2021). Universal multilingual NLU/ NLG models like mBERT (Devlin et al., 2019), mBART (Liu et al., 2020), XLM-R (Conneau et al., 2020) or mT5 support some major Indian languages. With IndicBART, we also intend to support many more Indic languages and utilize the relatedness between Indian languages (Kunchukuttan and Bhattacharyya, 2020; Goyal et al., 2020a) to learn better multilingual representations.

The following are the highlights of the IndicBART model:

- Supports English and 11 Indian languages including 7 Indo-Aryan (Assamese, Ben-
gali, Gujarati, Hindi, Marathi, Odiya, Punjabi) and 4 Dravidian (Kannada, Malayalam, Tamil, Telugu) languages. Of these languages, mBART25, mBART50 and mT5 support only 2, 7 and 9 languages respectively.

- The model is trained on IndicCorp dataset\(^1\) (Kakwani et al., 2020) which include large, high-quality news crawls for Indian languages as well as English content from Indian websites - thus being representative of Indian English and topics.

- The model is trained with denoising autoencoder objective following the mBART architecture. Since many Indic languages use Brahmi-derived scripts, we utilize this orthographic similarity (Kunchukuttan et al., 2018) to map all the Indic language data to a single script to achieve better cross-lingual transfer while using a compact vocabulary.

- The model size is 244M parameters which is smaller than the universal models such as mBART50 and mT5(-base) which contain 611M and 580M parameters respectively.

- We finetune IndicBART on two downstream generation tasks: machine translation and extreme summarization (Narayan et al., 2018). Our results are competitive or better than fine-tuning on larger models such as mBART50. Our analysis reveals that multilingual fine-tuning and single-script models enable good performance with smaller, but language-focused language models. In fact, script unification enables better performance than bilingual translation models even on languages not included in the pre-training stage.

- The models are available under an MIT license to spur further innovation in generation tasks for Indic languages.

2 Related Work

The work presented in this paper is related to natural language generation for Indian languages. In particular, we deal with issues such as pre-training, language family specific NLP, low-resource multilingual machine translation and summarization.

Pre-training has been extensively explored in recent years as it can help improve the performance of natural language processing applications. A vast majority of the work on pre-training is for natural language understanding whereas research on pre-training for natural language generation has only become popular recently. Models such as mT5 (Xue et al., 2021), BART (Lewis et al., 2020), mBART25 (Liu et al., 2020), mBART50 (Tang et al., 2020) etc. have shown to be tremendously useful in improving NLG performance in low-resource settings, especially for applications other than translation. However, most of these models do not focus on any language family in particular and are particularly unwieldy due to their large number of parameters. While language family specific NLU language models (Kakwani et al., 2020; Khanuja et al., 2021) and NMT models (Tan et al., 2019), we believe ours is the first effort to create a pre-trained NLG model for a specific language family. Previous work has shown the importance of language relatedness for cross-lingual transfer (Dabre et al., 2017; Aharoni et al., 2019; Kudugunta et al., 2019). In the case of Indic languages, orthographic similarity between Indic languages has been utilized to represent data in a common script and improve cross-lingual transfer (Dabre et al., 2018; Goyal et al., 2020b; Khemchandani et al., 2021).

With regards to low-resource machine translation, there has been a large body of work where the issue of poor translation quality has been tackled using transfer learning (Zoph et al., 2016; Chu et al., 2017; Dabre et al., 2019), pre-trained models (Liu et al., 2020), back-translation (Sennrich et al., 2016), unsupervised learning (Artetxe et al., 2018) and multilingualism (Dabre et al., 2020; Firat et al., 2016a,b). Our work focuses on closely related languages which often benefit from the possibility of shared vocabulary (Nguyen and Chiang, 2017). Indeed, methods using script unification have been particularly successful and we endeavour to develop models that follow such techniques. We do not focus on backtranslation in this work and it is complementary to our work (Liu et al., 2020).

It has been shown that pre-trained models significantly improve abstractive summarization quality (Lewis et al., 2020; Rothe et al., 2020). However, research on summarization has been limited by the unavailability of datasets, especially for Indic languages. Fortunately, recent work on multilingual summarization (Hasan et al., 2021) has enabled experimentation for a number of Indic languages.

\(^1\)https://indicnlp.ai4bharat.org/corpora
which is one of the focus of this paper. Existing works do not focus on script unification for summarization and our work intends to bridge this knowledge gap.

3 IndicBART

We describe the IndicBART model and its training in this section. The IndicBART model is conceptually based on the mBART-25/50 model family i.e. the model is trained on monolingual corpora with masked span reconstruction objective. We refer the readers to the mBART literature (Lewis et al., 2020; Liu et al., 2020) for architectural details and highlight specific model details and the differences from the mBART50 training setup.

Some of the considerations that drove our model choices are:

- Keep the model size compact to accelerate training and fine-tuning. Given that we are catering to a smaller set of related languages, we think a smaller model will suffice our purpose and will be usable by a larger base of users.

- In addition to Indian languages, we include English since transfer-learning from English is a natural usecase, and English is widely used in the Indian subcontinent. We also use English content from the Indian subcontinent to reflect relevant content.

- Utilize the orthographic similarity between Indian languages. Many major Indic language use abugida scripts derived from the Brahmi script. The logical character set is highly overlapping, though each script has its own code-point range in the Unicode standard. We map all the data to the same script (we choose Devanagari), enabling us to achieve better transfer learning with a more compact vocabulary.

3.1 Dataset and Tokenization

We train the IndicBART model on the IndicCorp dataset (Kakwani et al., 2020) which contains (M=) 11 Indic languages and English. The Indic languages are: Assamese, Bengali, Gujarati, Hindi, Kannada, Malayalam, Marathi, Oriya, Punjabi, Tamil and Telugu. The corpora statistics are given in Table 1. We train the model on a total of approx. 450 million sentences and 9 billion tokens. All the Indic language data is represented in a single script

| Language | #sentences |
|----------|------------|
| as       | 1.39       |
| bn       | 39.9       |
| en       | 54.3       |
| gu       | 41.1       |
| hi       | 63.1       |
| kn       | 53.3       |
| ml       | 50.2       |
| mr       | 34.0       |
| or       | 6.94       |
| pa       | 29.2       |
| ta       | 31.5       |
| te       | 47.9       |

Table 1: Monolingual corpora statistics (in millions).

i.e. the Devanagari script using the IndicNLP library (Kunchukuttan, 2020). We use a vocabulary size of 64K subword learnt from raw text using SentencePiece (Kudo, 2018; Kudo and Richardson, 2018). The subword model is trained on randomly sampled 1M sentences from the IndicCorp for each language for a total of 12M sentences. The model is trained at the sentence-level, unlike the mBART50 model which is trained on contiguous fixed-size text chunks potentially spanning multiple sentences.

3.2 Model Details

We mask (p=)35% of the words in each sentence by randomly sampling a span length according to a Poisson distribution (λ = 3.5). We use (N=) 6 encoder and decoder layers with hidden and filter sizes of 1024 and 4096, respectively, and 16 attention heads. We use dropouts of 0.1 and label smoothing of 0.1. We use the Adam optimizer with a maximum learning rate of 0.001 and label decay of 0.00001. We use linear learning rate warmup and decay with 16,000 warmup steps. The model has been trained with the YANMTT toolkit (Dabre and Sumita, 2021) which is based on the mBART implementation of the HuggingFace Transformers library (Wolf et al., 2020). We use batch sizes of 4096 tokens and train for 750,000 iterations on 48 NVIDIA V-100 GPUs which corresponds to roughly 2 epochs and took around 5 days. ²

²https://github.com/anoopkunchukuttan/indic_nlp_library

³https://github.com/prajdabre/yanmtt

⁴We could have trained for longer but we were limited by the availability of a large number of GPUs simultaneously.
4 Downstream Task: NMT

Machine Translation is a standard, popular, cross-lingual generation task on which various pretrained models are typically evaluated. We compare IndicBART with mBART50 which should be the most directly comparable model. We study their performance in: (a) low-resource scenarios, (b) in-domain and general domain settings, (c) multilingual training settings.

4.1 Datasets

Training For a low-resource setting, we use the PMI subset (Haddow and Kirefu, 2020) of the WAT 2021 MultiIndicMT\(^5\) (Nakazawa et al., 2021) training set for finetuning. This represents an extremely low-resource parallel corpus setting where we expect IndicBART to be the most helpful. We also experiment with extending the PMI data with the CVIT-PIB (henceforth PIB) data (Siripragrada et al., 2020) which is similar in domain to the former. We also use the large, general domain Samanantar corpus (Ramesh et al., 2021) to compare with the generalization capabilities of pretrained models which are finetuned with small corpora (PMI, PIB). Also note that the PMI and PIB data are included in the Samanantar data.

Testing We use the WAT 2021 MultiIndicMT testset and the FLORES101 devtest (Goyal et al., 2021) for evaluation of our models. Both these testsets are n-way parallel (2,390 and 1,012 sentences respectively). The WAT 2021 testset shares the same domain as the training set. The FLORES devtest comes from a different, general domain. We rely on the FLORES dataset to evaluate performance of models trained on the PMI/PIB domain on a more general domain.

Validation We use the WAT2021 development set to determine model convergence. Note that, we do not use the FLORES dev set and our validation set is drawn only from the PMI/PIB dataset. While this may hurt the final performance on FLORES, our objective is to determine the impact of fine-tuning a model on a specific domain and evaluating it for a more general domain.

The statistics of these corpora are given in Table 2. With this experimental setup, we aim to study the benefits of pre-training in low-resource settings (finetuned on PMI and PIB) and compare it with high-resource settings (trained on Samanantar).

| Pair | Low-resource | High-resource |
|------|--------------|---------------|
|      | PMI          | PIB           | Total         | Samanantar |
| bn-en| 23,306       | 91,985        | 115,291       | 8,435,620  |
| gu-en| 41,578       | 58,264        | 99,842        | 3,019,563  |
| hi-en| 50,349       | 266,545       | 316,894       | 8,467,772  |
| kn-en| 28,901       | -             | 28,901        | 4,014,931  |
| ml-en| 26,916       | 43,087        | 70,003        | 5,780,479  |
| mr-en| 28,974       | 114,220       | 143,194       | 3,288,874  |
| or-en| 31,966       | 94,494        | 126,460       | 990,466    |
| pa-en| 28,294       | 101,092       | 129,386       | 2,400,659  |
| ta-en| 32,638       | 115,968       | 148,606       | 5,095,763  |
| te-en| 33,380       | 44,720        | 78,100        | 4,775,516  |
| Total| 326,302      | 930,375       | 1,256,677     | 46,269,643 |

Table 2: Statistics of parallel corpora (#sentences).
| Model          | #Params | bn  | gu  | hi  | kn  | ml  | mr  | or  | pa  | ta  | te  |
|---------------|---------|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|
| XX-En         |         |     |     |     |     |     |     |     |     |     |     |
| Bi            | 78M     | 13.5| 27.4| 30.9| 22.5| 16.5| 18.4| 18.4| 27.1| 17.1| 16.5|
| M2O           | 78M     | 18.9| 24.8| 27.8| 23.8| 21.6| 20.7| 21.2| 26.4| 20.6| 21.8|
| MB50+Bi       | 611M    | 23.2| **35.4**| **38.3**| 26.8| **29.2**| 27.7| 27.8| **35.8**| 27.1| **30.8**|
| IB+M2O        | 244M    | **24.8**| 33.9| 37.2| **32.4**| 28.5| **28.5**| **28.8**| 35.7| **27.3**| 29.5|
| En-XX         |         |     |     |     |     |     |     |     |     |     |     |
| Bi            | 78M     | 4.5 | 17.9| 21.7| 12.1| 3.9 | 10.0| 9.2 | 17.9| 7.2 | 2.1 |
| O2M           | 78M     | 7.4 | 22.5| 25.9| 16.2| 5.6 | 14.7| 15.4| 21.9| 10.0| 2.7 |
| MB50+Bi       | 611M    | 8.6 | 23.5| 27.0| 17.4| 6.0 | 15.8| **11.6**| 24.5| 11.2| 3.3 |
| IB+O2M        | 244M    | **9.1**| 24.0| 27.3| **18.5**| 6.7 | **16.7**| 12.9 | 26.4| 11.6| 3.7 |

Table 3: Comparison of IndicBART with other models. Scores are reported on the WAT 2021 test set.

4.3 Models Trained

We train the following types models for our studies on IndicBART. Unless explicitly mentioned, our models are assumed to be trained/fine-tuned using the PMI training data.

1. Bi: Bilingual models trained from scratch.
2. MB50+Bi: Bilingual models that are fine-tuned on mBART50. Since mBART50 is not explicitly trained for Kannada, Punjabi and Oriya we map the scripts for these languages to Devanagari before fine-tuning. Results for these are italicized.
3. IB w/o SM+Bi: Bilingual models fine-tuned on the IndicBART model trained without script unification.
4. IB+Bi: Bilingual models fine-tuned on the IndicBART model trained with script unification.
5. M2O/O2M: Many-to-one or one-to-many models trained from scratch.
6. IB w/o SM+O2M/M2O: Many-to-one or one-to-many fine-tuned on the IndicBART model trained without script unification.
7. IB+O2M/M2O: Many-to-one or one-to-many fine-tuned on the IndicBART model trained with

checkpoint for a language pair with the highest development set BLEU score for that pair. Therefore, we treat multilingualism as a way to get a (potentially) different model per language pair leading to the best BLEU scores for that pair and not as a way to get a single model that gives the best performance for each language pair. During decoding the test sets, we use beam search with a beam of size 4 and a length penalty of 0.8. We report the BLEU scores on the decoded results computed using sacreBLEU. We consider the results of this model as our main result.

4.4.1 Main Results

Table 3 shows our main results where the models are trained on the PMI corpus and evaluated on the WAT21 testset. To summarize, IndicBART fine-tuned on all language pairs is significantly better than bilingual and multilingual baselines. It is also competitive or better than mBART50 tuned on a single language pair. IndicBART has only $\frac{1}{3}$rd the model size of mBART50, showing that it is advantageous to have compact models focusing on significantly fewer languages. IndicBART makes multilingual finetuning feasible with modest compute availability compared to multilingual finetuning of mBART50. Hence, for a fair comparison, we have only used bilingual finetuning of mBART50 as a baseline.

Table 4 contains the ablation tests giving the results for the impact of script unification and multilingual training when compared against using original scripts for languages and bilingual training. Comparing script unification against original script models, independent of multilingual or bilingual training, the former is clearly better than the latter which could indicate that script unification enables...
languages to better benefit from each other. On the other hand, multilingual training, independent of script unification, is definitely better than bilingual training. Ultimately, the combination of script unification and multilingual training tends to give the best results.

The case of Kannada, Punjabi and Oriya illustrates the utility of script unification. The results for these languages are italicized in the rows labelled MB50+Bi in Table 3. mBART50 was not pre-trained on these languages but we converted the training data in these languages in the Devanagari script\(^{10}\). With this trick, we still managed to get large performance improvements over the baselines and these improvements are often close to those exhibited by using IndicBART. This shows that we may not need to pre-train on all languages. However, explicitly training on the languages of interest should undoubtedly lead to the best translation quality.

### 4.4.2 Impact Of Corpora Size and Domain

Table 5 shows the impact of corpora sizes as well as training data domain on the final performance. In order to clearly evaluate the impact of domains, we evaluate on the WAT 2021 as well as the FLORES test sets. All models are one-to-many or many-to-one and are trained on unified script data. We fine-tune IndicBART on PMI (IB+PMI) as well as its combination with the PIB (IB+PMIPiB) dataset. For comparison we also train models on Samantar data from scratch (Samanantar) as well as by fine-tuning (IB+Samanantar) which represents models trained on a large amount of general domain data. We use the same model configuration as IndicBART making all 4 types of models identical in size.

Regardless of the test sets, comparing rows IB+PMI and IB+PMIPiB, it is clear that increasing the amount of fine-tuning data has a positive impact on the final translation quality. Even though the PMIPiB data is in-domain for the WAT 2021 test set and out-of-domain for the FLORES test set, the performance improvement on the latter is substantial. Furthermore, comparing rows IB+PMIPiB and Samantar, we can see widely different results depending on the test set. In the case of the WAT 2021 test set, fine-tuning on the PMIPiB dataset is comparable if not better than training on Samantar from scratch. This also shows that for domain specific models, having a small in-domain fine-tuning data is equivalent to if not better than having a large general domain data. On the other hand, we have the opposite observation on the FLORES test sets where training on Samantar data is clearly better. This makes sense because the PMI and PIB domains are very different from the domain of FLORES. PMI and PIB datasets form a tiny fraction of Samantar and thus there wont be any overfitting on the PMI and PIB domains by training on Samantar. Finally, the scores in the row IB+Samanantar show that the performance tends to degrades sometimes around 1 BLEU due to fine-tuning. This observation is in line with the ones by Liu et al. (2020) in resource rich settings.

We can conclude with a high degree of confidence that using IndicBART for fine-tuning will be more successful when evaluating on a specific domain using small domain specific training corpora. On the other hand, it will be better to use large domain specific corpora when working on specific

| Model         | bn | gu | hi | kn | ml | mr | or | pa | ta | te |
|---------------|----|----|----|----|----|----|----|----|----|----|
| IB+M2O       | 24.8 | 33.9 | 37.2 | 32.4 | 28.5 | 28.5 | 28.8 | 35.7 | 27.3 | 29.5 |
| IB w/o SM+M2O | 24.1 | 33.8 | 35.5 | 31.2 | 27.9 | 28.0 | 28.1 | 35.7 | 29.9 | 26.9 |
| IB+Bi        | 23.6 | 35.5 | 36.8 | 31.6 | 27.9 | 26.8 | 28.3 | 36.3 | 27.0 | 29.9 |
| IB w/o SM+Bi | 22.3 | 34.9 | 36.6 | 30.8 | 27.5 | 26.7 | 28.0 | 36.0 | 26.3 | 29.7 |

Table 4: Ablation studies to study the impact of multilingualism and script unification on downstream performance of IndicBART. Scores are reported on the WAT 2021 test set.
Table 5: Ablation study of the impact of using different sizes of fine-tuning corpora (PMI and its combination with PIB) and their comparison against a model trained from scratch as well as fine-tuned on a general domain corpus (Samanantar). We evaluate on the WAT 2021 as well as the FLORES test sets.

| Model          | Test Set: WAT 2021 | XX-En                  | Test Set: FLORES | XX-En                  |
|----------------|---------------------|------------------------|------------------|------------------------|
|                | bn  | gu  | hi  | kn  | ml  | mr  | or  | pa  | ta  | te  | bn  | gu  | hi  | kn  | ml  | mr  | or  | pa  | ta  | te  | bn  | gu  | hi  | kn  | ml  | mr  | or  | pa  | ta  | te  |
| IB+PMI         | 24.8 | 33.9 | 37.2 | 32.4 | 28.5 | 28.5 | 28.8 | 35.7 | 27.3 | 29.5 | 9.1  | 24.0 | 27.3 | 18.5 | 6.7  | 16.7 | 12.9 | 26.4 | 11.6 | 3.7  |
| IB+PMIPB       | 28.9 | 38.8 | 41.7 | 34.6 | 33.2 | 32.5 | 33.2 | 41.3 | 32.0 | 32.0 | 11.1 | 25.5 | 33.0 | 18.9 | 7.2  | 19.1 | 14.3 | 27.1 | 13.6 | 3.6  |
| Samanantar     | 27.9 | 39.0 | 41.8 | 34.8 | 32.7 | 32.0 | 32.9 | 41.4 | 31.2 | 34.4 | 9.7  | 24.7 | 33.0 | 17.5 | 7.0  | 18.4 | 13.3 | 25.5 | 12.7 | 5.8  |
| IB+Samanantar  | 27.1 | 38.0 | 41.0 | 34.1 | 31.6 | 31.1 | 32.3 | 40.1 | 30.1 | 32.4 | 9.4  | 24.2 | 33.0 | 17.2 | 6.5  | 17.7 | 13.5 | 25.6 | 11.8 | 5.6  |

Table 6: Evaluation of Nepali and Sinhala to English translation which IndicBART has not seen during pre-training.

| Model | ne-en | si-en |
|-------|-------|-------|
| Baseline | 5.2  | 4.3  |
| IB+Bi   | 10.5 | 8.5  |

4.4.3 Evaluating On Unseen Languages

Table 6 shows what happens when we perform fine-tuning for languages that IndicBART is not trained on. Note that for Sinhala we have to resort to script unification into Devanagari. The baselines, trained using the unified script IndicBART vocabulary will seem weaker than what is reported in previous work but it should be noted that the vocabulary was not actually trained for Nepali and Sinhala. Regardless, fine-tuning leads to substantial improvements in translation quality which indicates the utility of IndicBART even for unseen languages. We can expect some additional pre-training on monolingual corpora for these languages to have an even larger impact on fine-tuning performance.

5 Downstream Task: Summarization

We compare the performance of fine-tuning IndicBART and mBART50 on an abstractive summarization task for Indic languages textitviz the challenging extreme summarization task (Narayan et al., 2018). Given that summarization datasets are small, the utility of pre-training can be well-studied on this task.

5.1 Dataset

We used the multilingual XLSum dataset (Hasan et al., 2021), which has been curated from the same source as the English XSum dataset, for our experiments. The Indic languages we focus on for
evaluating our IndicBART models are: Bengali, Gujarati, Hindi, Marathi, Punjabi, Tamil and Telugu. We use the official splits which differ from the splits in Hasan et al. (2021). The corpora statistics are given in Table 7.

5.2 Model Training Settings

Different from NMT, we used YANMTT for fine-tuning IndicBART and the official summarization fine-tuning scripts created by HuggingFace for fine-tuning mBART50.

5.2.1 Fine-tuning IndicBART

We only perform multilingual fine-tuning with 8 GPUs. We use maximum document-summary lengths of 512-64 tokens which loosely follows previous work (Lewis et al., 2020). Unlike NMT, we do not train models from scratch as they would not work given the small data sizes and difficulty of summarization. We use dropouts of 0.1, label smoothing of 0.1, learning rate warmup steps of 4,000, learning rate of 0.001 and weight decay of 0.00001 with the ADAM optimizer. We train our models till convergence on the development set Rouge scores (Rouge-L F1) (Lin, 2004) for all languages which are computed via greedy decoding every 1,000 batches. Similar to NMT, we save the best performing checkpoints on the development set Rouge scores (Rouge-L F1) (Lin, 2004) for all languages which are computed via greedy decoding every 1,000 batches. We report Rouge scores on the decoded results computed using multilingual Rouge scoring toolkit. We report Rouge scores on the decoded results computed using multilingual Rouge scoring toolkit. We report Rouge scores on the decoded results computed using multilingual Rouge scoring toolkit.

5.2.2 Fine-tuning mBART50

We only perform monolingual fine-tuning on 1 GPU using the scripts provided in the HuggingFace transformer implementation. We use default tuning parameters in the provided scripts. Except for the following, other settings are the same as for fine-tuning IndicBART. We use label smoothing of 0.1, a learning rate of 0.00005 for mBART50, weight decay of 0.01 with the AdamW optimizer and 10% of the total number of steps as learning rate warmup. We train for 10 epochs and use the Rouge-L F1 metric for selecting the best model.

5.3 Models Trained

1. MB: A mBART50 model fine-tuned for a given language. Data for languages (Punjabi) not present in mBART50 have their scripts mapped to Devanagari and then used for fine-tuning. Results for these are italicized.

2. IB w/o SM: Our separate script IndicBART model fine-tuned multilingually for all Indic languages in consideration.

3. IB: Same as 2 but with a unified script IndicBART model and unified script training data for summarization.

5.4 Results

Table 8 contains the results for the summarization experiments. We first compare IndicBART with mBART50 and then discuss the impact of script unification.

5.4.1 IndicBART vs mBART50

Comparing the columns, MB and IB shows that our unified script IndicBART model, IB, is at par with if not better than mBART50. mBART50 is trained on document level data and one would expect that it would strongly outperform IndicBART but the latter gives competitive results despite being trained on sentence level data. In addition, IndicBART contains less than half the number of parameters as mBART50. The performance in case of Punjabi, Gujarati, Marathi and Telugu is noteworthy as they are higher than mBART50.

5.4.2 Impact of Script Unification

Comparing, columns IB w/o SM and IB shows that script unification has a positive impact in almost

---

Table 7: Statistics of the Indic portion of the multilingual XL-Sum dataset (Hasan et al., 2021) that we used for training our summarization models.

| Language | Train | Dev  | Test  |
|----------|-------|------|-------|
| bn       | 8,102 | 1,012| 1,012 |
| gu       | 9,119 | 1,139| 1,139 |
| hi       | 70,778| 8,847| 8,847 |
| mr       | 10,903| 1,362| 1,362 |
| pa       | 8,215 | 1,026| 1,026 |
| ta       | 16,222| 2,027| 2,027 |
| te       | 10,421| 1,302| 1,302 |

---

11[https://github.com/csebuetnlp/xl-sum/](https://github.com/csebuetnlp/xl-sum/)

12This means that 4-grams wont be repeated in the output.

13[https://github.com/csebuetnlp/xl-sum/tree/master/multilingual_rouge_scoring](https://github.com/csebuetnlp/xl-sum/tree/master/multilingual_rouge_scoring)

14Multilingual fine-tuning on these large models is time consuming and hence impractical.

15[https://github.com/huggingface/transformers/tree/master/examples/pytorch/summarization](https://github.com/huggingface/transformers/tree/master/examples/pytorch/summarization)
Table 8: Results for Indic languages in XL-Sum dataset (Hasan et al., 2021). We compare between mBART50(MB), IndicBART (IB) and IndicBART without script unification (IB w/o SM) models. Evaluation is conducted using ROUGE metric. Scores in italics for a language are the ones obtained via mapping scripts to Devanagari as the corresponding model was not explicitly trained for that language.

all cases. Given that Indic languages are related to each other, script unification enables better transfer learning especially during multilingual fine-tuning. We would also like the reader to consider the results for Punjabi using mBART50. mBART50 is not trained on Punjabi and so we convert the Punjabi script to Devanagari and fine-tune it. Despite not having seen any Punjabi training data, the mBART50 model can be successfully used for Punjabi summarization which shows that, just like in the case of NMT, existing pre-trained models can be easily retrofitted to serve the needs for unseen languages via simple script unification. We have already observed this in the case of NMT where Nepali and Sinhala translation into English worked well despite the IndicBART model not having seen either of these Indic languages. In the future we plan to study the summarization quality of Nepali and Sinhala datasets part of XL-Sum (Hasan et al., 2021), using IndicBART via script unification.

6 Conclusion and Future Work

We present IndicBART, a multilingual, pre-trained sequence-to-sequence model for Indic languages to support development of NLG applications for Indic languages. IndicBART supports 11 Indic languages and English, and utilizes the orthographic similarity of Indic scripts to enable better cross-lingual transfer. Our experiments on multilingual fine-tuning IndicBART for NMT and summarization show that the model is competitive with or better than those obtained using existing large models such as mBART50. Due to the use of orthographic similarity, the model can be used to build translation models for languages like Sinhala and Nepali that are not included in pre-training. We show that script unification has a strong positive impact on translation and summarization. We also showed that IndicBART, thanks to its script independent nature, can be readily used for enabling translation for languages such as Sinhala and Nepali which IndicBART has not been explicitly trained for.

As next steps, we plan to add support for more Indic languages in IndicBART, starting with all the 22 languages listed in the 8th schedule of the Indian constitution. Increased language coverage and lower compute demands can democratize access to NLP technologies. We plan to train both larger and smaller models on longer text chunks (documents) and larger text corpora. We would also like to incorporate advances in multilingual pre-trained models, cross-lingual transfer and cross-lingual tasks for Indic languages.

References

Roee Aharoni, Melvin Johnson, and Orhan Firat. 2019. Massively multilingual neural machine translation. In Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers), pages 3874–3884, Minneapolis, Minnesota. Association for Computational Linguistics.

Mikel Artetxe, Gorka Labaka, Eneko Agirre, and Kyunghyun Cho. 2018. Unsupervised neural machine translation. In Proceedings of International Conference on Learning Representations, Vancouver, BC, Canada.

Chenhui Chu, Raj Dabre, and Sadao Kurohashi. 2017. An empirical comparison of domain adaptation methods for neural machine translation. In Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Pa-
Raj Dabre, Chenhui Chu, and Anoop Kunchukuttan. 2020. A survey of multilingual neural machine translation. *ACM Comput. Surv.*, 53(5).

Raj Dabre, Atsushi Fujita, and Chenhui Chu. 2019. Exploiting multilingualism through multistage fine-tuning for low-resource neural machine translation. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 1410–1416, Hong Kong, China. Association for Computational Linguistics.

Raj Dabre, Anoop Kunchukuttan, Atsushi Fujita, and Eiichiro Sumita. 2018. Nict’s participation in wat 2018: Approaches using multilingualism and recurrently stacked layers. In *Proceedings of the 32nd Pacific Asia Conference on Language, Information and Computation: 5th Workshop on Asian Translation: 5th Workshop on Asian Translation*.

Raj Dabre, Tetsuji Nakagawa, and Hideto Kazawa. 2017. An empirical study of language relatedness for transfer learning in neural machine translation. In *Proceedings of the 31st Pacific Asia Conference on Language, Information and Computation*, pages 282–286. The National University (Philippines).

Raj Dabre and Eiichiro Sumita. 2021. Yanmtt: Yet another neural machine translation toolkit.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Orhan Firat, Kyunghyun Cho, and Yoshua Bengio. 2016a. Multi-way, multilingual neural machine translation with a shared attention mechanism. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 866–875, San Diego, California. Association for Computational Linguistics.

Orhan Firat, Baskaran Sankaran, Yaser Al-Otaibian, Fatos T. Yarman Vural, and Kyunghyun Cho. 2016b. Zero-resource translation with multi-lingual neural machine translation. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 268–277, Austin, Texas. Association for Computational Linguistics.

Naman Goyal, Cynthia Gao, Vishrav Chaudhary, Peng-Jen Chen, Guillaume Wenzek, Da Ju, Sanjana Krishnan, Marc’Aurelio Ranzato, Francisco Guzmán, and Angela Fan. 2021. The FLORES-101 evaluation benchmark for low-resource and multilingual machine translation. *CoRR*, abs/2106.03193.

Vikrant Goyal, Anoop Kunchukuttan, Rahul Kejriwal, Siddharth Jain, and Amit Bhagwat. 2020a. Contact relatedness can help improve multilingual NMT: Microsoft STCI-MT @ WMT20. In *Proceedings of the Fifth Conference on Machine Translation*, pages 202–206, Online. Association for Computational Linguistics.

Vikrant Goyal, Anoop Kunchukuttan, Rahul Kejriwal, Siddharth Jain, and Amit Bhagwat. 2020b. Contact Relatedness can help improve multilingual NMT: Microsoft STCI-MT @ WMT20. In *Conference on Machine Translation*.

Barry Haddow and Faheem Kirefu. 2020. PMIndia – A Collection of Parallel Corpora of Languages of India. *arxiv* 2001.09907.

Tahmid Hasan, Abhik Bhattacharjee, Md. Saiful Islam, Kazi Mubasshar, Yuan-Fang Li, Yong-Bin Kang, M. Sohel Rahman, and Rifat Shahriyar. 2021. XLsum: Large-scale multilingual abstractive summarization for 44 languages. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 4693–4703, Online. Association for Computational Linguistics.

Divyanshu Kakwani, Anoop Kunchukuttan, Satish Golla, Gokul N.C., Avik Bhattacharyya, Mitesh M. Khapra, and Pratyush Kumar. 2020. IndicNLPSuite: Monolingual corpora, evaluation benchmarks and pre-trained multilingual language models for Indian languages. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 4948–4961, Online. Association for Computational Linguistics.

Mihir Kale and Abhinav Rastogi. 2020. Text-to-text pre-training for data-to-text tasks. In *Proceedings of the 13th International Conference on Natural Language Generation*, pages 97–102, Dublin, Ireland. Association for Computational Linguistics.

Simran Khana, Diksha Bansal, Sarvesh Mehtani, Savya Khosla, Atreyee Dey, Balaji Gopalan, Dilip Kumar Margam, Pooja Aggarwal, Rajiv Teja Nagipogu, Shachi Dave, Shruti Gupta, Subhash Chandra Bose Gali, Vish Subramanian, and Partha Talukdar. 2021. Muril: Multilingual representations for indian languages.
Taku Kudo. 2018. Subword regularization: Improving word candidates. In Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers), pages 1312–1323, Online. Association for Computational Linguistics.

Taku Kudo and John Richardson. 2018. SentencePiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. In Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations, pages 66–71, Brussels, Belgium. Association for Computational Linguistics.

Anoop Kunchukuttan and Pushpak Bhattacharyya. 2020. Utilizing Language Relatedness to improve Machine Translation: A Case Study on Languages of the Indian Subcontinent. arXiv preprint arXiv:2003.08925.

Anoop Kunchukuttan, Mitesh Khpaura, Gurneet Singh, and Pushpak Bhattacharyya. 2018. Leveraging orthographic similarity for multilingual neural transliteration. Transactions of the Association for Computational Linguistics, 6:303–316.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, pages 7871–7880, Online. Association for Computational Linguistics.

Chin-Yew Lin. 2004. ROUGE: A package for automatic evaluation of summaries. In Text Summarization Branches Out, pages 74–81, Barcelona, Spain. Association for Computational Linguistics.

Yinhan Liu, Jitao Gu, Naman Goyal, Xian Li, Sergey Edunov, Marjan Ghazvininejad, Mike Lewis, and Luke Zettlemoyer. 2020. Multilingual denoising pre-training for neural machine translation. Transactions of the Association for Computational Linguistics, 8:726–742.

Toshiaki Nakazawa, Hideki Nakayama, Chenchen Ding, Raj Dabre, Shohei Higashiyama, Hideya Mino, Isao Goto, Win Pa Pa, Anoop Kunchukuttan, Shantipriya Parida, Ondřej Bojar, Chenhui Chu, Akiko Eriguchi, Kaori Abe, and Sadao Oda, Yusuke Kurohashi. 2021. Overview of the 8th workshop on Asian translation. In Proceedings of the 8th Workshop on Asian Translation, Bangkok, Thailand. Association for Computational Linguistics.

Shashi Narayan, Shay B. Cohen, and Mirella Lapata. 2018. Don’t give me the details, just the summary! topic-aware convolutional neural networks for extreme summarization. In Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, pages 1797–1807, Brussels, Belgium. Association for Computational Linguistics.

Toan Q. Nguyen and David Chiang. 2017. Transfer learning across low-resource, related languages for neural machine translation. In Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 2: Short Papers), pages 296–301, Taipei, Taiwan. Asian Federation of Natural Language Processing.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. BLEU: a method for automatic evaluation of machine translation. In Proceedings of the 40th Annual Meeting on Association for Computational Linguistics, pages 311–318.

Romain Paulus, Caiming Xiong, and Richard Socher. 2018. A deep reinforced model for abstractive summarization. In International Conference on Learning Representations.

Matt Post. 2018. A call for clarity in reporting BLEU scores. In Proceedings of the Third Conference on Machine Translation: Research Papers, pages 186–191, Brussels, Belgium. Association for Computational Linguistics.

Ratish Puduppully, Li Dong, and Mirella Lapata. 2019. Data-to-text generation with content selection and planning. In The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, The Thirty-First Innovative Applications of Artificial Intelligence Conference, IAAI 2019, The Ninth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019, Honolulu, Hawaii, USA, January 27 - February 1, 2019, pages 6908–6915. AAAI Press.
Gowtham Ramesh, Sumanth Doddapaneni, Aravinh Bheemaraj, Mayank Jobanputra, Raghuvan AK, Ajitesh Sharma, Sujit Sahoo, Harshita Diddee, Mahalakshmi J, Divyanshu Kakwani, Navneet Kumar, Aswin Pradeep, Kumar Deepak, Vivek Raghuvan, Anoop Kunchukuttan, Pratyush Kumar, and Mitesh Shantadevi Khapra. 2021. Samanantar: The largest publicly available parallel corpora collection for 11 indic languages. CoRR, abs/2104.05596.

Sascha Rothe, Shashi Narayan, and Aliaksei Severyn. 2020. Leveraging pre-trained checkpoints for sequence generation tasks. Transactions of the Association for Computational Linguistics, 8:264–280.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Improving neural machine translation models with monolingual data. In Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 86–96, Berlin, Germany. Association for Computational Linguistics.

Shashank Siripragrada, Jerin Philip, Vinay P. Namboodiri, and C V Jawahar. 2020. A multilingual parallel corpora collection effort for Indian languages. In Proceedings of The 12th Language Resources and Evaluation Conference, pages 3743–3751, Marseille, France. European Language Resources Association.

Xu Tan, Yi Ren, Di He, Tao Qin, and Tie-Yan Liu. 2019. Multilingual neural machine translation with knowledge distillation. In International Conference on Learning Representations (ICLR), New Orleans, LA, USA.

Yuqing Tang, Chau Tran, Xian Li, Peng-Jen Chen, Naman Goyal, Vishrav Chaudhary, Jiatao Gu, and Angela Fan. 2020. Multilingual translation with extensible multilingual pretraining and finetuning.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-art natural language processing. In Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations, pages 38–45, Online. Association for Computational Linguistics.

Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V. Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, Jeff Klingner, Apurva Shah, Melvin Johnson, Xiaobing Liu, Lukasz Kaiser, Stephan Gouws, Yoshikiyo Kato, Taku Kudo, Hideto Kazawa, Keith Stevens, George KJurian, Nishant Patil, Wei Wang, Cliff Young, Jason Smith, Jason Riesa, Alex Rudnick, Oriol Vinyals, Greg Corrado, Macduff Hughes, and Jeffrey Dean. 2016. Google’s neural machine translation system: Bridging the gap between human and machine translation. CoRR, abs/1609.08144.

Linting Xue, Noah Constant, Adam Roberts, Mihir Kale, Rami Al-Rfou, Aditya Siddhant, Aditya Barua, and Colin Raffel. 2021. mT5: A massively multilingual pre-trained text-to-text transformer. In Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pages 483–498, Online. Association for Computational Linguistics.

Barret Zoph, Deniz Yuret, Jonathan May, and Kevin Knight. 2016. Transfer learning for low-resource neural machine translation. In Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, pages 1568–1575, Austin, Texas. Association for Computational Linguistics.