Aksharantar: Open Indic-language Transliteration datasets and models for the Next Billion Users

Yash Madhani[1] **Sushane Parthan**[2] **Priyanka Bedekar**[3] **Gokul NC**[4]
Ruchi Khapra[5] **Anoop Kunchukuttan**[6] **Pratyush Kumar**[7] **Mitesh M. Khapra**[8]
AI4Bharat[1,2,3,4,5,6,7,8] IIT Madras[1,2,3,6,7,8] Microsoft[6,7]
[1,2,3]`{cs20s002,cs20d201,cs20m050}@smail.iitm.ac.in`
[4]`gokulnc@ai4bharat.org` [5]`jain.ruchi03@gmail.com`
[6,7]`{ankunchu,pratykumar}@microsoft.com` [8]`miteshk@cse.iitm.ac.in`

Abstract

Transliteration is very important in the Indian language context due to the usage of multiple scripts and the widespread use of romanized inputs. However, few training and evaluation sets are publicly available. We introduce *Aksharantar*[1], the largest publicly available transliteration dataset for Indian languages created by mining from monolingual and parallel corpora, as well as collecting data from human annotators. The dataset contains **26 million transliteration pairs** for **21 Indic languages** from **3 language families** using **12 scripts**. Aksharantar is **21 times** larger than existing datasets and is the first publicly available dataset for **7 languages** and **1 language family**. We also introduce the Aksharantar testset comprising **103k word pairs** spanning **19 languages** that enables a fine-grained analysis of transliteration models on native origin words, foreign words, frequent words, and rare words. Using the training set, we trained *IndicXlit*, a multilingual transliteration model that improves accuracy by 15% on the Dakshina test set, and establishes strong baselines on the Aksharantar testset introduced in this work. The models, mining scripts, transliteration guidelines, and datasets are available at `https://github.com/AI4Bharat/IndicXlit` under open-source licenses.

1 Introduction

The Indian subcontinent is home to diverse languages across four major language families written in multiple scripts (Daniels and Bright, 1996). In various settings such as instant messaging, web search, and social media, these languages are commonly romanized owing to users' familiarity with the input tools for the Roman script. Often, there is a large diversity in how words are romanized: For instance, even the short word मैं (*I*) can be romanized in multiple ways: *main, mai, mein, mei* which overlap with the ways of romanizing another short word, में (*in*). This widespread usage of romanization and lack of standardization implies that accurate transliteration models form a critical component in the NLP stack for Indian languages used by over 735 million Internet users (KPMG and Google, 2017). Further, accurate transliteration models have been shown to improve machine translation (Durrani et al., 2014b), romanized language models (Khanuja et al., 2021), NER (Klementiev and Roth, 2006), and script unification for multilingual models (Muller et al., 2021).

Given its importance, a transliteration for Indian languages has received considerable research focus (Kumaran et al., 2010; Chen et al., 2018b; Kunchukuttan et al., 2018a; Roark et al., 2020). However, the *state-of-the-art* (SOTA) results as reported in Roark et al. (2020) on the Roman script to Indian transliteration task have relatively low top-1 accuracy values ranging between 33.2% to 67.6% with an average of 51.8% across 12 languages. We believe that the low accuracy is a result of limited training datasets that are not representative of the diverse variations of romanization. We aim to address this open challenge in a manner similar to recent advances for low-resource languages in machine translation (Ramesh et al., 2022; Costa-jussà et al., 2022) and speech recognition (Bhogale et al., 2022; Radford et al., 2022), namely by mining massive training corpora from web-scale data.

Specifically, we create Aksharantar, a transliteration corpora from Roman script to 12 Indic language scripts spanning 21 Indic languages. It is

[1]meaning *transliteration* in Sanskrit

All language codes are ISO 639-2					Aksharantar trainset (in thousands)						Aksharantar (AK) testset				
Language	**Code**	**Script**	**Family**	**Examples**	**Exs**	**Wik**	**Sam**	**Ind**	**Man**	**Total**	**Freq**	**Uni**	**NEF**	**NEI**	**Tot**
Assamese	asm	Bengali	IA	ভাৰত	-	2	3	203	19	**217**	1,690	1,938	742	1,161	**5,531**
Bengali	ben	Bengali	IA	ভারত	104	107	193	1,115	14	**1,337**	1,071	1,198	1,059	1,681	**5,009**
Bodo	brx	Devanagari	ST	भारत	-	-	-	36	13	**44**	1,119	1,143	729	1,145	**4,136**
Gujarati	guj	Gujarati	IA	ભારત	111	8	67	1,096	21	**1,236**	2,725	2,521	1,005	1,517	**7,768**
Hindi	hin	Devanagari	IA	भारत	234	44	289	1,149	49	**1,522**	1,726	1,924	826	1,217	**5,693**
Kannada	kan	Kannada	DR	ಭಾರತ್	51	<1	69	2,930	27	**3,010**	1,851	2,361	877	1,307	**6,396**
Kashmiri	kas	Perso-Arabic	IA	بھارت	-	<1	-	35	37	**64**	3,095	2,588	816	1,208	**7,707**
Konkani	kok	Devanagari	IA	भारत	65	-	-	619	37	**702**	1,531	1,536	817	1,209	**5,093**
Maithili	mai	Devanagari	IA	भारत	102	7	-	252	42	**370**	1,892	1,591	819	1,210	**5,512**
Malayalam	mal	Malayalam	DR	ഭാരത്	61	1	59	4,097	30	**4,195**	2,261	2,596	835	1,219	**6,911**
Manipuri	mni	Meetei Mayek	ST	ꯚꯥꯔꯠ	-	-	-	12	11	**16**	2,754	-	886	1,285	**4,925**
Marathi	mar	Devanagari	IA	भारत	60	26	49	1,486	49	**1,594**	2,091	2,375	831	1,276	**6,573**
Nepali	nep	Devanagari	IA	भारत	-	10	-	2,455	6	**2,458**	1,058	1,049	817	1,209	**4,133**
Oriya	ori	Oriya	IA	ଭାରତ	-	1	23	380	13	**398**	1,068	1,153	821	1,214	**4,256**
Punjabi	pan	Gurmukhi	IA	ਭਾਰਤ	78	21	104	481	13	**611**	1,049	1,144	858	1,265	**4,316**
Sanskrit	san	Devanagari	IA	भारत	-	3	-	1,860	38	**1,881**	1,411	1,515	976	1,432	**5,334**
Sindhi	snd	Perso-Arabic	IA	ڀارت	39	<1	-	53	-	**82**	-	-	-	-	-
Sinhala	sin	Sinhala	IA	භාරත්	42	-	-	-	-	**37**	-	-	-	-	-
Tamil	tam	Tamil	DR	பாரத்	71	1	61	3,202	14	**3,301**	1,467	1,141	828	1,246	**4,682**
Telugu	tel	Telugu	DR	భారత్	97	<1	82	2,416	14	**2,521**	1,105	1,135	947	1,380	**4,567**
Urdu	urd	Perso-Arabic	IA	بھارت	111	<1	-	649	3	**748**	-	2,437	817	1,209	**4,463**
Total	-	-	-	-	**1,225**	**229**	**1,000**	**24,525**	**451**	**26,345**	**30,964**	**31,345**	**16,306**	**24,390**	**103,005**

Table 1: Summary of the Aksharantar dataset. Languages for which there are no existing datasets are shown in **bold**. The language families represented are IA (Indo-Aryan), DR (Dravidian) and ST (Sino-Tibetan). All examples shown are transliterations of word "Bharat". The Aksharantar trainset covers 21 Indic languages, while the testset covers 19 Indic languages. The trainset sources are Exs (existing), Wik (Wikidata), Sam (Samanantar), Ind (IndicCorp) and Man (manual transliterations). Total is sum of unique word pairs. The Aksharantar testset has multiple subsets defined as AK-Freq (most frequent words), AK-Uni (uniformly sampled words), AK-NEF (foreign named entities) and AK-NEI (Indian named entities).

21 times larger than existing publicly available datasets and includes 7 new languages (*Assamese, Bodo, Kashmiri, Manipuri, Nepali, Oriya, Sanskrit*) and *1* new language family (*Sino-Tibetan*) for which no transliteration corpora wasa available previously. The parallel transliteration corpora have been mined from Wikidata (Vrandečić and Krötzsch, 2014), Samanantar parallel translation corpora (Ramesh et al., 2022), and IndicCorp monolingual corpora (Doddapaneni et al., 2022) along with a compilation of existing transliteration corpora. In addition, the corpora contain a diverse set of native language words that have been transliterated manually to ensure coverage of native words of different lengths, different n-gram characteristics, and infrequent words - characteristics that mined corpora lack.

Our next major contribution is the *Aksharantar* testset, an evaluation benchmark dataset for romanized transliteration. Table 1 shows the statistics of the benchmark set, which comprises *103K* word pairs spanning *19* languages. The benchmark contains (a) native language words with diverse n-gram characteristics, and (b) named entities of Indic and foreign origin spanning different entity categories. Most publicly available testsets focus on named entities (Chen et al., 2018a), but the representation of native words is important for input tools. While the Dakshina testset (Roark et al., 2020) represents native words, they include only the most frequent words in Wikipedia - which is not representative of all native words. Our testset ensures greater diversity in native language word coverage. Our experiments confirm that our testset is indeed more diverse and challenging, hence making it more suitable for better evaluation of transliteration models. It is known that transliteration of English-origin and native-origin words have their own distinct behavior (Ahmed et al., 2011; Khapra and Bhattacharyya, 2009), hence we create testsets for both word classes to enable this fine-grained evaluation of transliteration models.

Our next contribution is a multilingual model for romanized to native script transliteration for Indian languages. *Our model gives SOTA performance on the Dakshina benchmark (Roark et al., 2020) for all the 12 languages in common with Dakshina showing an improvement of 15% in accuracy over previous results.* It also establishes a strong baseline on the Aksharantar testset.

Our final contribution is a detailed analysis of the model's performance on the rich Aksharantar testset. Ablation studies indicate that the increased data size as well as the manually collected diverse dataset is a major contributor to the improved performance. The fine-grained testsets reveal named entities and low-frequency words as areas for improving transliteration models.

The code and models are available under an MIT license[2], the Aksharantar benchmark and all data we created manually are available under the CC-BY license[3], whereas all the mined data is available under the CC0 license[4].

2 Related Work

Existing Indic Transliteration Corpora. Very few transliteration corpora exist with Indian language-Roman script transliterations. Refer to Appendix A for detailed listing and statistics. Most significant among these are the Dakshina dataset (Roark et al., 2020) and the BrahmiNet corpus (Kunchukuttan et al., 2015). Dakshina contains native language words sourced from Wikipedia and their romanizations created by native speakers, unlike Aksharantar mostly consists of commonly used and shorter Indic language words.

Mining Transliteration Pairs. Irvine et al. (2010) mine the name pairs from Wikipedia using inter-language links between pages in multiple languages (similar to our use of Wikidata titles' multilingual information). Some approaches mine transliteration pairs from comparable document pairs based on a variety of heuristic signals (Klementiev and Roth, 2006; Udupa et al., 2008, 2009). Sajjad et al. (2012) proposed a generative model for efficient unsupervised/semi-supervised mining of transliteration pairs. We employ unsupervised mining method proposed by Sajjad et al. (2012) to mine transliteration pairs from parallel corpora. Richardson et al. (2013) mine transliteration pairs from monolingual corpora by transliterating the vocabulary of one language using a baseline system and then by filtering the generated data.

Multilingual Models. Multilingual models have been shown to improve performance on low-resource languages for many NLP tasks by transfer from high-resource languages and aligning representation of multiple languages in the same vector space (Johnson et al., 2017; Conneau et al., 2020). The transfer could be between genetically-related languages (Nguyen and Chiang, 2017) or contact-related languages (Goyal et al., 2020). Multilingual models have been explored successfully for different NLP tasks involving Indian languages, such as language representation modeling (Kakwani et al., 2020; Dabre et al., 2022), machine translation (Ramesh et al., 2022; Dabre et al., 2018), POS tagging (Plank et al., 2016; Khemchandani et al., 2021) and named-entity recognition (Murthy and Bhattacharyya, 2016; Mhaske et al., 2022). Our models are closest to Kunchukuttan et al. (2018b) and Kunchukuttan et al. (2021), who propose multilingual training for transliteration using LSTM-based models, focusing on transliterations involving orthographically similar languages. In this work, we train Transformer-based models using much larger training sets.

[2]https://opensource.org/licenses/MIT

[3]https://creativecommons.org/licenses/by/4.0

[4]https://creativecommons.org/publicdomain/zero/1.0

Types of errors	Examples
leaked translation word pairs	अंतर्संयुक्त → Interconnected
highly agglutinated words on one side	અંકલેશ્વરનો → Ankleshwar

Table 2: Examples of incorrect mined pairs from translation corpora.

3 Mining Transliteration pairs

We explore sources for mining transliteration pairs. First, we compile existing publicly-available transliteration corpora listed in Appendix A. Then, we explore large-scale mining of transliterations from Wikidata, parallel translation corpora and monolingual corpora.

3.1 Mining from Wikidata

Wikidata (Vrandečić and Krötzsch, 2014) is a multilingual, structured database containing items that are either entities, things, concepts, or terms. Each entity has *labels* that are common names of the items in multiple languages. We restrict ourselves to person and location entities since their labels will be transliterations. We extract such English-Indian language label pairs creating transliteration pairs. Appendix B provides more details on Wikidata mining. The candidate pairs are filtered using a transliteration validator described in Section 4.3.

3.2 Mining from Translation Corpora

Parallel sentences can contain transliteration pairs in the form of named entities, loan words, and cognates. To mine these transliteration pairs, we first use an off-the-shelf word-aligner *GIZA++* (Och and Ney, 2003) with the default settings to learn the word alignments between parallel sentences. These aligned words can either be translations or transliterations. Then, we use the unsupervised method suggested by Sajjad et al. (2012), as implemented in the transliteration module (Durrani et al., 2014a) of *Moses* (Koehn et al., 2007), to mine transliteration pairs from these word alignments by distinguishing transliterations and non-transliterations. Please refer to Appendix C for more details. Using this approach, we mine transliteration pairs from the *Samanantar* parallel corpora (v0.3) (Ramesh et al., 2022), the largest publicly available parallel corpora for Indian languages when we started this project. The above-mentioned process can result in some wrong transliteration pairs being mined (see Table 2). To filter out such pairs, we use a rule-based transliteration validator (described in Section 4.3) which checks the correctness of consonant alignment between transliteration pairs and works well for the kinds of erroneous transliteration pairs mined by the above-mentioned method.

3.3 Mining from Monolingual Corpora

Monolingual corpora often contain borrowed words from other languages (particularly English). We mine transliteration pairs between English and Indian languages using only a list of words in the source and target languages. We first train multilingual transliteration models with the same setting described in Section 5 using available data (data from existing sources and mined from parallel translation corpora) in both directions (M_{ex}: en $\rightarrow$ Indic and M_{xe}: Indic $\rightarrow$ en). We use the IndicCorp dataset (Doddapaneni et al., 2022) to create a list of words for English and Indic languages (L_x). Given the word w_x in L_x, we generate its transliteration (w_e') using the M_{xe} model (*e.g.,* जर्मिनेट $\rightarrow$ germinat). We find similar new English words (w_e) from the English word list such that there exist at least three common character 4-grams between w_e' and w_e (*e.g.,* germinated, germinate, germinating, germinates). The candidate pair (w_x, w_e) is scored using models in both directions.

$$s(w_x, w_e) = \frac{1}{2}\{M_{xe}(w_x, w_e) + M_{ex}(w_e, w_x)\}$$

We retain candidate transliteration pairs with score (average character-level log probability in both directions) greater than a threshold $t{=}{-}0.35$, which was set after our analysis of transliteration pairs across languages (*e.g.,* germinated, germinate, germinating, germinates).

3.4 Quality of the mined data

To validate the quality of the mined corpora, we perform a human evaluation on a subset of mined pairs. We randomly sampled 500 Indic-Roman script-mined pairs equally from IndicCorp and Samanantar corpora in 12 languages. Two passes of validation by different language validators were performed on this data. Annotators were asked to mark the pairs which were valid transliterations. The *accuracy* of mining is defined to be the percentage of valid pairs in the subset that was manually judged. We achieved minimum accuracy of 80% per language and average accuracy of 89% across all 12 languages. The results of human evaluation, summarised in Table 3, show that data mined from Samanantar and IndicCorp has high accuracy.

We analyzed the pairs judged as invalid and found that they included the following errors:
Vowel errors: a/e being added incorrectly at the end of transliterations, missing vowels, and wrong usage of vowels (*e.g.,* अमिताभ $\rightarrow$ Amtabha [missing 'i' after 'm' and added 'a' at the end]).
Suffix errors: Suffixes wrongly transliterated or missed altogether, leading to partial transliterations (*e.g.,* रोनाल्डोही$\rightarrow$ Ronaldo, टोकिया$\rightarrow$ Tokyo).

We found that most erroneous pairs were partial transliterations which introduce limited albeit useful noise in the training data. The results of the human judgment and qualitative analysis confirm the high quality of mined transliteration pairs which makes it useful for training transliteration models.

4 Manual Data Collection

Collating existing sources and mining transliterations from web sources is insufficient for building a representative transliteration dataset because (i) mined corpora are predominantly composed of named entities, (ii) romanized native words in the Dakshina dataset only cover frequent words in languages occurring on Wikipedia and may not ensure sufficient word diversity to account for various transliteration phenomena (since Wikipedia for most Indic languages is small), (iii) mined data only covers 12 languages for which sufficient monolingual/parallel corpora are available and which have high grapheme-to-phoneme correspondence making mining feasible, (iv) we still need a diverse and accurate standard testset for all Indic languages.
To address these needs, we collect transliteration

Dataset	asm	ben	guj	hin	kan	kok	mai	mal	mar	pan	san	tam
Ind	90.8	92.8	90.8	96.8	98.0	98.8	90.8	94.0	96.8	94.8	78.0	80.0
Sam	92.8	92.0	84.0	76.0	80.0	-	-	80.0	90.0	86.0	84.0	80.0

Table 3: Transliteration mining accuracy on human-judged samples of Ind (IndicCorp Corpora) and Sam (Samanantar Corpora).

pairs in 19 Indic languages from trained annotators across India. This section describes the data collection process wherein (i) Indic words to be romanized are selected to ensure diversity and coverage across languages, and (ii) high-quality, manually curated romanizations for these Indic words are collected at scale by setting up a systematic process to ensure quality control and annotator productivity on a digital platform. We collect multiple romanizations for native script words to capture the variations in romanization of native words. Words in Indic scripts have a more standardized orthography. Our data collection protocol ensures that we can collect diverse romanizations to train our transliteration models.

4.1 Sourcing Indic words

Words for manual transliteration in 19 languages were sourced from publicly available datasets. We use IndicCorp (Doddapaneni et al., 2022) to source Indic language words for 11 languages (*Assamese, Bengali, Gujarati, Hindi, Kannada, Malayalam, Marathi, Oriya, Punjabi, tamil* and *Telugu*). For 6 languages (*Maithili, Konkani, Bodo, Nepali, Kashmiri* and *Urdu*) we use the LDC-IL corpus (Choudhary, 2021). We collect *Sanskrit* words from religious scriptures such as the Mahabharata (Sukthankar, 2017), while for *Manipuri* we use Wikipedia. We ensure that these source words are not already covered in the sources mentioned in Section 3. We select native script words for manual transliteration with the goal of ensuring coverage of varying length words, diverse n-grams, common as well as infrequent words, and foreign origin words. We use a combination of the following methods for selecting diverse source words:

Most frequent words: To account for the most frequent words in a language, we select the top 5000 words for each language.

N-gram Diversity: We train a 4-gram character LM over all words for each language using KenLM with Kneser-Ney smoothing (Heafield, 2011; Heafield et al., 2013), whose probabilities are a good indicator of 4-gram frequencies in a given word. We compute log probability scores (normalized by word length and scaled to 0-1 range) for each candidate word using the character LM. The words are then sharded into bins corresponding to the 10 probability deciles. Words are uniformly sampled from each bin, ensuring n-gram diversity in source words, complementing mined corpora which are mostly composed of named entities and head inputs. We sampled a total of 10,000 words per language using this method.

Named Entities: We sampled 2000 named entities in English spanning 3 broad categories: names, locations, and organizations, covering Indian and foreign origin words. We sourced Indian and foreign personal names and locations by randomly sampling words from collections on websites dedicated for the same. Organization names are sourced from the stock market library list of 1600+ companies listed in NSE[5]. These 2000 named entities consist of 800 names (400 each of Indian and foreign origin), 800 locations (400 each of Indian and foreign origin), and 400 Indian organizations.

4.2 Annotation Process and QC

We collect transliterations via a two-step process akin to a maker-checker process. A human annotator creates multiple romanized variants for a native word. To aid transliterators, we provide an automatic rule-based transliteration validator. The automatic validator flags potentially wrong transliterations, helping the transliterator correct mistakes made while entering word variants. The correctness of variants is checked by a human *validator*, who also has the freedom to enter unique word variants. Through multiple pilot projects, we studied different annotation styles, identified common annotation errors made, and devised a set of basic instructions. Annotators were free to enter all common variants while following the instructions as much as possible. Due to budget constraints, the maximum number of variants is capped to 4 per transliterator and 2 per validator. We keep all vari-

[5]https://www.samco.in/knowledge-center/articles/nse-listed-companies/

ants in Roman script for a given Indic word and create (Roman,Indic) script pairs for all of the variants. More details regarding annotators, annotation instructions, *etc.* are described in Appendix D.

4.3 Automatic Validation

To aid transliterators, we provide an automatic rule-based transliteration validator. The tool flags potentially wrong transliterations, helping the transliterator correct mistakes made while entering word variants. Typically, we found that the transliteration validator helped identify typographical errors and other mistypings and ensured consistency in transliterations. Note that this automatic validator only serves as a guide to transliterators, who can override its checks at their discretion. The transliteration validator is based on the Transliteration Equivalence algorithm for English (Roman script)-Hindi described in Khapra et al. (2014) which checks equivalence of the consonant mappings in a potential transliteration pair. More details are described in Appendix E. In total, we collect 554k transliteration pairs across 19 languages, which are split into 451k pairs for the training set pairs and 103k pairs for the test set.

5 IndicXlit: A Multilingual Model for Transliteration

With the Aksharantar training set, we train a transliteration model, *IndicXlit*, for transliterating romanized Indic language input to the native script. IndicXlit is a single multilingual, multi-script transliteration model that supports 21 Indic languages. We train a joint model since: (a) low-resource languages would benefit from transfer learning, (b) previous works show that multilingual transliteration models are better at generating canonical spellings (Kunchukuttan et al., 2018a), and (c) maintenance is easier for a single model.

Model Architecture. We use a transformer-based encoder-decoder architecture (Vaswani et al., 2017). It is a multilingual character level transliteration model (Kunchukuttan et al., 2021) in a one-to-many setting, which consumes a romanized character sequence and generates an output character sequence in the Indic language script. The input sequence includes a special *target language tag* token to specify the target language (Johnson et al., 2017). Model vocabulary, hyper-parameters, and training details are described in Appendix F. The model size is **11** million parameters.

Decoding We use beam search with beam size $=$ 4. In addition, we also re-rank top-4 candidates using a revised score F_c generated by combining 2 features, (i) a word-level unigram LM score (P_c), (ii) transliteration score (character-level log probability) (T_c) as shown.

$$F_c = \alpha T_c + (1 - \alpha) P_c \qquad (1)$$

We use $\alpha = 0.9$ based on tuning the parameter on the development set.
Table 5 shows the statistics of the train and validation splits used to train IndicXlit.

6 Analysis of IndicXlit quality

We analyze IndicXlit's transliteration quality on the Dakshina and Aksharantar testset. We strictly ensure that there is no word overlap between training and test/validation sets for inference. Note that the testsets considered for overlap computation include the Dakshina testset. We remove a pair *(en, t)* from the training set if (i) the Roman script word *en* is present in the romanized validation/test set of any language pair, or (ii) the Indic script word *t* is present in the Indic language validation/test set of any language pair. We report top-1 word level accuracy as our primary evaluation metric (Chen et al., 2018a). Additionally, we report top-3 and top-5 accuracies as well as F1-score in Appendix G as our secondary evaluation metrics. We observe that major trends on all metrics are consistent.

6.1 Quality on Dakshina testset

We compare IndicXlit with the best reported results on the Dakshina testset (in Table 4). Note that the Dakshina testset covers only 12 of the languages that are part of the Aksharantar dataset. The IndicXlit model substantially improves the results reported by Roark et al. (2020) on the Dakshina dataset, with a 15% improvement in average accuracy across languages. Since the size of training data is a major difference between the two models, it is clear that large-scale mined transliteration pairs help to substantially improve the transliteration quality. Multilingual training also helps improve the transliteration quality. These observations are further supported by ablation results reported in Section 7. The largest improvements are seen for *mar* (30.3%) and *guj* (25.7%), possibly because they are similar to the high resource *hin* language and *mar* also shares the script with Hindi. The least improvements are seen for *tam* (4.6%) and *tel* (8.9%).

Model	ben	guj	hin	kan	mal	mar	pan	snd	sin	tam	tel	urd	avg
Roark et al. (2020)	49.4	49.5	50.0	66.2	58.3	49.7	40.9	33.2	54.7	65.7	67.6	36.7	51.8
Our models trained on Dakshina dataset													
Monolingual	41.8	42.7	46.7	58.3	52.8	41.4	37.3	35.0	52.4	56.0	63.2	34.7	46.9
Multilingual	47.2	51.0	51.8	66.4	56.5	51.0	42.2	41.3	58.7	63.5	67.1	38.3	52.9
IndicXlit	**55.4**	**62.0**	**60.5**	**77.1**	**63.5**	**64.8**	**47.2**	**48.5**	**63.9**	**68.1**	**73.3**	**42.1**	**60.5**

Table 4: Top-1 accuracies reported on the Dakshina test set. We trained the monolingual and multilingual models on the Dakshina dataset using the same architecture as IndicXlit, so the impact of the dataset can be isolated.

Split	asm	ben	brx	guj	hin	kan	kas	kok	mai	mal	mni	mar	nep	ori	pan	san	snd	sin	tam	tel	urd	Total
Training	179	1,231	36	1,143	1,299	2,907	47	613	283	4,101	10	1,453	2,397	346	515	1,813	60	32	3,231	2,430	699	24,823
Validation	4	11	3	12	6	7	4	4	4	8	3	8	3	3	9	3	8	4	9	8	12	133

Table 5: Training and validation set statistics for Aksharantar. All numbers are in thousands.

6.2 Quality on Aksharantar testset

We report the results of IndicXlit on the Aksharantar testset in Table 6, particularly looking at the accuracy on various sub-testsets to understand model performance on different categories of words.

Frequent words are easier. The performance on the Dakshina dataset and the AK-Freq dataset, both comprised of frequent words in the language, is similar. The AK-Freq testset has the best performance across all subtestsets, suggesting that this test set is easiest to transliterate. These words are shorter on average and might be comprised of common n-grams - explaining the good performance.

Words with diverse n-grams are harder. On the other hand, the AK-Uni testset comprised of uniformly sampled words with diverse n-gram characteristics is much more challenging, with average accuracy being 10 points lower than the AK-Freq testset. This testset presents a challenging usecase for transliteration systems. Lower accuracy on this testset can be attributed to the average length of words and the rarity of the n-grams.

Named entities are the hardest. Named entity testsets are the most difficult testsets even though named entities constitute a large fraction of mined training data. Given the larger grapheme-phoneme mismatch for foreign entities, lower performance on this set is not surprising. While performance on Indian named entities is better than that on foreign named entities, their transliteration accuracy is still lower than the uniformly sampled test set. This is surprising and warrants further investigation.

Some languages are harder. In terms of language-wise accuracy, the lowest-performing languages are ones using the Arabic script (*urd, kas*) or those with lesser training data (*asm, brx, ori*).

Re-ranking helps on average. Unigram re-ranking of the candidates helps improve the transliteration accuracy substantially by 12% on an average across languages (See Table 6 for results). LM re-ranking mostly benefits the native language words and high resource languages with a lot of monolingual data for training LMs.

Re-ranking doesn't help for named entities. Unigram re-ranking shows limited benefits for named entities. This is not surprising since named entities might not be well represented in the LM given their rarity. Similarly, low-resource languages with limited monolingual data benefit less from LM re-ranking. Infrequent words thus pose a challenge to the quality of transliteration models.

6.3 Error analysis

To understand the errors made by IndicXlit, we analysed the output of the model for 100 randomly sampled words each for *Bengali, Gujarati, Hindi, Kannada, Marathi, Punjabi, Telugu* from the Dakshina dataset. The most common errors across languages are with respect to vowels (60%) and similar consonants (25%), while other errors (15%) include *viz. gemination, acronyms, contextual ambiguity, valid alternatives, and language-specific errors*. Appendix H provides a more detailed discussion of the error analysis with examples.

7 Ablation Studies

We describe various ablation studies carried out on Dakshina testset for 9 languages *viz. Bengali, Gujarati, Hindi, Kannada, Malayalam, Marathi, Punjabi, Tamil, Telugu* and their results which drove

Testset	**asm**	**ben**	**brx**	**guj**	**hin**	**kan**	**kas**	**kok**	**mai**	**mal**	**mni**	**mar**	**nep**	**ori**	**pan**	**san**	**tam**	**tel**	**urd**	**avg**
Without re-ranking																				
Dakshina	-	55.4	-	62.0	60.5	77.1	-	-	-	63.5	-	64.8	-	-	47.2	-	68.1	73.3	42.1	61.4
AK-Freq	65.9	63.0	74.8	65.3	58.6	80.6	31.2	65.3	78.6	71.6	83.1	74.6	80.1	66.7	49.0	81.5	73.7	90.0	-	69.7
AK-Uni	55.1	60.4	66.7	58.1	52.9	72.6	27.8	61.1	64.3	58.6	-	54.0	79.9	51.9	32.1	75.9	64.6	79.3	48.3	59.1
AK-NEF	38.9	36.4	30.8	45.6	55.8	53.2	13.2	27.2	33.3	29.4	44.6	49.5	49.1	29.6	31.1	19.5	39.0	53.5	48.0	38.3
AK-NEI	39.1	40.5	30.8	51.5	61.4	48.7	25.0	39.5	49.5	37.8	44.6	56.6	55.4	32.1	40.1	26.7	44.6	51.5	47.6	43.3
Micro-avg	52.8	54.0	52.3	60.4	58.3	72.0	26.1	51.7	61.2	59.2	66.8	62.5	66.7	45.4	43.8	56.4	63.9	71.7	43.8	56.3
After re-ranking top 4 candidates with $\alpha = 0.9$																				
Dakshina	-	69.4	-	73.8	72.4	85.2	-	-	-	73.5	-	76.1	-	-	60.4	-	78.5	84.4	46.9	72.1
AK-Freq	77.4	79.7	78.4	84.5	67.9	90.5	30.0	76.2	87.5	83.3	91.7	85.4	86.6	79.2	60.4	90.0	85.8	94.7	-	79.4
AK-Uni	67.2	69.2	65.0	68.7	63.0	82.1	27.1	58.9	62.7	69.4	-	63.3	82.7	63.5	42.6	88.0	76.5	86.0	46.1	65.7
AK-NEF	37.0	36.3	28.9	47.4	59.1	56.4	13.1	30.0	35.8	30.5	42.9	51.7	55.6	28.4	34.1	18.6	42.9	55.9	51.8	39.8
AK-NEI	41.3	43.1	29.4	54.1	67.9	52.2	28.8	42.5	55.7	41.1	44.7	61.8	62.4	33.2	42.7	29.3	47.6	55.2	52.7	46.6
Micro-avg	60.7	65.9	52.0	72.1	68.1	79.9	26.2	55.4	65.5	68.5	71.5	72.1	72.3	51.8	54.7	62.9	73.5	80.2	47.4	63.2

Table 6: Top-1 accuracy for IndicXlit on various testsets.

No	**Description**	**ben**	**guj**	**hin**	**kan**	**mal**	**mar**	**pan**	**tam**	**tel**	**avg**
Impact of various transliteration sources (monolingual models)											
(1)	Dakshina baseline	41.8	42.7	46.7	58.3	52.8	41.4	37.3	56.0	63.2	**48.9**
(2)	(1)+Existing	41.9	43.0	48.6	58.9	51.4	43.4	38.7	58.5	65.1	**50.0**
(3)	(2)+Wikidata	44.2	43.9	49.0	57.7	50.3	45.8	40.0	57.1	63.8	**50.2**
(4)	(3)+Samanantar	48.4	47.4	53.1	64.1	55.6	49.0	40.1	62.1	67.7	**54.2**
(5)	(4)+IndicCorp	56.0	60.0	56.3	76.3	64.8	65.4	46.0	67.7	73.3	**62.9**
(6)	(5)+Manual	56.0	59.1	58.4	76.8	62.7	64.6	45.4	65.7	74.1	**62.5**
Impact of multilinguality and script unification (baseline: (5))											
(7)	Multi-script	54.9	60.8	58.8	76.7	64.0	64.2	47.6	67.4	73.1	**63.1**
(8)	Single-script	55.4	61.9	58.2	77.5	64.8	65.2	47.3	68.2	73.4	**63.5**
Impact of language family specific models (baseline: (7))											
(9)	(IA & DR) models	56.7	61.9	59.5	77.5	64.6	65.5	48.2	68.6	73.8	**64.0**

Table 7: Top-1 accuracies from experiments in the ablation study. For (9) we train two language family specific models, Indo-Aryan languages (IA: ben, guj, hin, mar, pan) and Dravidian languages (DR: kan, mal, tam, tel)

Model	**asm**	**ben**	**guj**	**hin**	**kan**	**kok**	**mai**	**mal**	**mar**	**nep**	**ori**	**pan**	**tam**	**tel**	**urd**	**avg**
Monolingual	24.7	50.9	56.2	54.5	71.1	38.3	36.7	**58.9**	**60.2**	14.7	24.0	42.9	62.9	**71.9**	31.8	46.7
Multilingual	**29.3**	**51.6**	**57.2**	**56.0**	**71.6**	**44.9**	**52.7**	58.9	60.2	**44.2**	**27.7**	**43.9**	**63.4**	71.2	**38.8**	**51.4**

Table 8: Monolingual *vs.* multilingual models (micro-averaged accuracy over all testsets).

Dataset	**ben**	**guj**	**hin**	**kan**	**mal**	**mar**	**pan**	**tam**	**tel**	**avg**
All	**54.1**	**58.5**	**56.6**	**71.9**	57.9	59.9	41.9	61.1	**72.0**	**59.3**
No manual	50.9	38.3	54.5	71.1	**58.9**	**60.2**	**42.9**	**62.9**	71.9	56.9

Table 9: Impact of manually collected pairs (micro-averaged accuracy over all testsets).

the design choices of the IndicXlit model described in Section 5. Results of the following research questions are presented in Table 7.

Impact of various transliteration corpora sources. We train separate monolingual models for each language. We initially trained a baseline model by using only the Dakshina training set, followed by successive addition of transliteration pairs collected/mined from various sources. We observe a consistent increase in transliteration quality as transliteration pairs from various sources are added. Particularly, we observe a substantial improvement in performance when we add word pairs mined from monolingual corpora, IndicCorp, which constitutes the largest component of Aksharantar. The addition of manually collected transliteration pairs does not have an impact on these languages and the Dakshina testset since IndicCorp already contains sufficient data to model the frequent words that are part of the Dakshina testset. However, as shown in Table 9, we observe that manually collected data improves the micro-averaged transliteration accuracy over Dakshina and all Aksharantar testsets, *viz.* AK-Freq, AK-Uni, AK-NEF, AK-NEI. This suggests that manually collected data improves accuracy on other testset categories. Moreover, manual data is necessary for extremely low-resource languages with no data in the public domain and for bootstrapping transliteration mining efforts.

Impact of Multilingual Models. We see that multilingual models show a slight improvement over monolingual results on the Dakshina benchmark. In another experiment, we compare monolingual and multilingual models (for *18* languages) using all sources (except manually collected datasets) and observe a substantial increase in accuracy for low-resource languages using multilingual models (Table 8). Thus multilingual models substantially improve performance for low-resource languages, while at least retaining performance on high-resource languages with a single model.

Impact of script unification. Indic scripts have a unique Unicode codepoint range, a 1-1 mapping between most characters of different scripts is possible since the Unicode standard accounts for similarities between Indic scripts. This can potentially improve transfer learning between languages. We experiment with *single script models* converting characters from all Brahmi-derived scripts to Devanagari scripts using the IndicNLP library (Kunchukuttan, 2020). A special language token is added to every input sequence to distinguish the original Indic language, as described in Section 5. After decoding, the Devanagari script output is converted back to the target language's Indic script using the 1-1 mapping. We observe that single-script and multi-script models have similar performance. As these models are already trained on an already ample amount of data, the single-script model doesn't provide additional benefits from transfer learning in addition to multilingual representation learning. Given the small difference and negligible model size overhead, we opt to use a multi-script model for all Indic languages to simplify the pre-processing of data and incorporation of scripts such as the Arabic script, which cannot be easily mapped to the Devanagari script.

Impact of language family specific models. We observe that language-family specific models are slightly better than a pan-Indic model. Given the small difference in quality and the convenience of maintaining and deploying a single model, we choose to train IndicXlit as a pan-Indic model.

Impact of word-level unigram LM re-ranking. We observe 12% improvement in accuracy by re-ranking the top-4 candidates. This gain is over and above the 15% gain obtained by using the Aksharantar training set.

8 Conclusion

In this work, we take a major step toward creating publicly available, open datasets and open-source models for transliteration in Indic languages. We introduce Aksharantar, the largest transliteration parallel corpora for 21 languages containing 26 million transliteration pairs, and covering 20 of 22 languages listed in the Indian constitution. We also create a diverse, high-quality testset for romanized to Indic script transliteration, covering word pairs with various characteristics and enabling fine-grained analysis of different transliteration use-cases. We also build IndicXlit, a transformer-based transliteration model, for romanized input to Indic script transliteration. IndicXlit achieves state-of-the-art results on the Dakshina testset. We also provide baseline results on the new Aksharantar testset along with a qualitative analysis of the model performance.

Limitations

The benchmark for transliteration for the most part contains clean words (grammatically correct, single script, etc.). Data from the real world might be noisy (ungrammatical, mixed scripts, code-mixed, invalid characters, etc.). A better representative benchmark might be useful for such use cases. However, the usecases captured by this benchmark should suffice for the collection of clean transliteration corpora. This also represents a first step for many low-resource languages where no transliteration benchmark exists.

In this work, training data is limited to the 20 languages and test data is limited to 19 languages listed in the 8^{th} schedule of the Indian constitution. Further work is needed to extend the benchmark to many more widely used languages in India (which has about 30 languages with more than a million speakers). Subsequent to the acceptance of this work, we have also released training and testsets for one more Indic language *viz.* Dogri (*doi*) which are available on the project website.

In this work, we describe word-level testsets. However, the typical usecase for transliteration is keyboard input of sentences (or at least a sequence of words). In such cases, the context would be useful to improve transliteration. A sentence-level transliteration benchmark would be useful for evaluation such contextual transliteration models. The Dakshina dataset has sentence-level transliteration testsets for 12 languages. In a project concurrent to this work (Madhani et al., 2023), we have created sentence-level transliteration testsets for 22 Indic languages.

In this work, we have only explored romanized to native script transliteration. However, there is a need for native script to romanized models as well for processing romanized Indic language text that is also prevalent on the web. Subsequent to the acceptance of this work, we have also released an Indic to Roman script IndicXlit model trained on the Aksharantar corpus. This model is also available on the project website.

Ethics Statement

For the human annotations on the dataset, the language experts are native speakers of the languages from the Indian subcontinent. We collaborated with external agencies for the annotation task. The payment was based on their skill set and experience, determined by the external agencies, and adhered to the government's norms. The dataset is free from harmful content. The annotators were made aware of the fact that the annotations would be released publicly and the annotations contain no private information. The proposed benchmark builds upon existing datasets. These datasets and related works have been cited.

The annotations are collected on a publicly available dataset and will be released publicly for future use.

Acknowledgements

We thank EkStep Foundation and Nilekani Philanthropies for their generous grant towards building datasets, models, tools and resources for Indic languages. We would like to thank the Ministry of Electronics and Information Technology of the Government of India for their generous grant through the Digital India Bhashini project. We also thank the Centre for Development of Advanced Computing for providing compute time on the Param Siddhi Supercomputer. We also thank Microsoft for their grant to support research on Indic languages. We would like to thank Vivek Seshadri and Karya team for supporting us with using the Karya platform for data collection. We would also like to thank all the language experts who created the manual transliterations for the training and testsets.

References

Basil Abraham, Danish Goel, Divya Siddarth, Kalika Bali, Manu Chopra, Monojit Choudhury, Pratik Joshi, Preethi Jyothi, Sunayana Sitaram, and Vivek Seshadri. 2020. Crowdsourcing speech data for low-resource languages from low-income workers. In *Proceedings of The 12th Language Resources and Evaluation Conference, LREC 2020, Marseille, France, May 11-16, 2020*, pages 2819–2826. European Language Resources Association.

Umair Z Ahmed, Kalika Bali, Monojit Choudhury, and VB Sowmya. 2011. Challenges in designing input method editors for indian lan-guages: The role of word-origin and context. In *Proceedings of the Workshop on Advances in Text Input Methods (WTIM 2011)*, pages 1–9.

Naveen Arivazhagan, Ankur Bapna, Orhan Firat, Dmitry Lepikhin, Melvin Johnson, Maxim Krikun, Mia Xu Chen, Yuan Cao, George F. Foster, Colin Cherry, Wolfgang Macherey, Zhifeng Chen, and Yonghui Wu. 2019. Massively multilingual neural machine translation in the wild: Findings and challenges. *CoRR*, abs/1907.05019.

Lei Jimmy Ba, Jamie Ryan Kiros, and Geoffrey E. Hinton. 2016. Layer normalization. *CoRR*, abs/1607.06450.

Joseph Benjamin and N C Gokul. 2020. Ai4bharat storyweaver xlit dataset. `https://github.com/AI4Bharat/IndianNLP-Transliteration.git`.

Kaushal Santosh Bhogale, Abhigyan Raman, Tahir Javed, Sumanth Doddapaneni, Anoop Kunchukuttan, Pratyush Kumar, and Mitesh M Khapra. 2022. Effectiveness of mining audio and text pairs from public data for improving asr systems for low-resource languages. *arXiv preprint arXiv:2208.12666*.

Nancy Chen, Xiangyu Duan, Min Zhang, Rafael E Banchs, and Haizhou Li. 2018a. Whitepaper on news 2018 shared task on machine transliteration. In *Proceedings of the Seventh Named Entity Workshop. Association for Computational Linguistics*.

Nancy F. Chen, Rafael E. Banchs, Min Zhang, Xiangyu Duan, and Haizhou Li. 2018b. Report of NEWS 2018 named entity transliteration shared task. In *Proceedings of the Seventh Named Entities Workshop, NEWS@ACL 2018, Melbourne, Australia, July 20, 2018*, pages 55–73. Association for Computational Linguistics.

Manu Chopra, Indrani Medhi-Thies, Joyojeet Pal, Colin Scott, William Thies, and Vivek Seshadri. 2019. Exploring crowdsourced work in low-resource settings. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems, CHI 2019, Glasgow, Scotland, UK, May 04-09, 2019*, page 381. ACM.

Narayan Lal Choudhary. 2021. LDC-IL: The Indian repository of resources for language technology. *Language Resources and Evaluation*, pages 1–13.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Unsupervised cross-lingual representation learning at scale. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8440–8451, Online. Association for Computational Linguistics.

Marta R Costa-jussà, James Cross, Onur Çelebi, Maha Elbayad, Kenneth Heafield, Kevin Heffernan, Elahe Kalbassi, Janice Lam, Daniel Licht, Jean Maillard, et al. 2022. No language left behind: Scaling human-centered machine translation. *arXiv preprint arXiv:2207.04672*.

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*, Hong Kong. Association for Computational Linguistics.

Raj Dabre, Himani Shrotriya, Anoop Kunchukuttan, Ratish Puduppully, Mitesh M. Khapra, and Pratyush Kumar. 2022. IndicBART: A Pre-trained Model for Natural Language Generation of Indic Languages. In *Findings of the Association for Computational Linguistics*.

Peter T Daniels and William Bright. 1996. *The world's writing systems*. Oxford University Press on Demand.

Sumanth Doddapaneni, Rahul Aralikatte, Gowtham Ramesh, Shreya Goyal, Mitesh M Khapra, Anoop Kunchukuttan, and Pratyush Kumar. 2022. IndicXTREME: A multi-task benchmark for evaluating indic languages. *arXiv preprint arXiv:2212.05409*.

Nadir Durrani, Barry Haddow, Philipp Koehn, and Kenneth Heafield. 2014a. Edinburgh's phrase-based machine translation systems for WMT-14. In *Proceedings of the Ninth Workshop on Statistical Machine Translation, WMT@ACL 2014, June 26-27, 2014, Baltimore, Maryland, USA*, pages 97–104. The Association for Computer Linguistics.

Nadir Durrani, Hassan Sajjad, Hieu Hoang, and Philipp Koehn. 2014b. Integrating an unsupervised transliteration model into statistical machine translation. In *Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics, volume 2: Short Papers*, pages 148–153.

Vikrant Goyal, Anoop Kunchukuttan, Rahul Kejriwal, Siddharth Jain, and Amit Bhagwat. 2020. 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.

Kenneth Heafield. 2011. KenLM: Faster and smaller language model queries. In *Proceedings of the Sixth Workshop on Statistical Machine Translation*, pages 187–197, Edinburgh, Scotland. Association for Computational Linguistics.

Kenneth Heafield, Ivan Pouzyrevsky, Jonathan H. Clark, and Philipp Koehn. 2013. Scalable modified Kneser-Ney language model estimation. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 690–696, Sofia, Bulgaria. Association for Computational Linguistics.

Dan Hendrycks and Kevin Gimpel. 2016. Bridging nonlinearities and stochastic regularizers with gaussian error linear units. *CoRR*, abs/1606.08415.

Ann Irvine, Chris Callison-Burch, and Alexandre Klementiev. 2010. Transliterating from all languages. In *Proceedings of the 9th Conference of the Association for Machine Translation in the Americas: Research Papers*.

Melvin Johnson, Mike Schuster, Quoc V. Le, Maxim Krikun, Yonghui Wu, Zhifeng Chen, Nikhil Thorat, Fernanda Viégas, Martin Wattenberg, Greg Corrado, Macduff Hughes, and Jeffrey Dean. 2017. Google's multilingual neural machine translation system: Enabling zero-shot translation. *Transactions of the Association for Computational Linguistics*, 5:339–351.

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.

Simran Khanuja, Diksha Bansal, Sarvesh Mehtani, Savya Khosla, Atreyee Dey, Balaji Gopalan, Dilip Kumar Margam, Pooja Aggarwal, Rajiv Teja Nagipogu, Shachi Dave, et al. 2021. Muril: Multilingual representations for indian languages. *arXiv preprint arXiv:2103.10730*.

Mitesh Khapra and Pushpak Bhattacharyya. 2009. Improving transliteration accuracy using word-origin detection and lexicon lookup. In *Proceedings of the 2009 Named Entities Workshop: Shared Task on Transliteration (NEWS 2009)*, pages 84–87, Suntec, Singapore. Association for Computational Linguistics.

Mitesh M. Khapra, Ananthakrishnan Ramanathan, Anoop Kunchukuttan, Karthik Visweswariah, and Pushpak Bhattacharyya. 2014. When transliteration met crowdsourcing : An empirical study of transliteration via crowdsourcing using efficient, non-redundant and fair quality control. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation, LREC 2014, Reykjavik, Iceland, May 26-31, 2014*, pages 196–202. European Language Resources Association (ELRA).

Yash Khemchandani, Sarvesh Mehtani, Vaidehi Patil, Abhijeet Awasthi, Partha Talukdar, and Sunita Sarawagi. 2021. Exploiting language relatedness for low web-resource language model adaptation: An Indic languages study. 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.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Alexandre Klementiev and Dan Roth. 2006. Weakly supervised named entity transliteration and discovery from multilingual comparable corpora. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, pages 817–824, Sydney, Australia. Association for Computational Linguistics.

Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Chris Dyer, Ondrej Bojar, Alexandra Constantin, and Evan Herbst. 2007. Moses: Open source toolkit for statistical machine translation. In *ACL 2007, Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics, June 23-30, 2007, Prague, Czech Republic*. The Association for Computational Linguistics.

KPMG and Google. 2017. Indian languages, defining india's internet. Report link.

A Kumaran, Mitesh M. Khapra, and Haizhou Li. 2010. Report of NEWS 2010 transliteration mining shared task. In *Proceedings of the 2010 Named Entities Workshop*, pages 21–28, Uppsala, Sweden. Association for Computational Linguistics.

Anoop Kunchukuttan. 2020. The IndicNLP Library. `https://github.com/anoopkunchukuttan/indic_nlp_library/blob/master/docs/indicnlp.pdf`.

Anoop Kunchukuttan, Siddharth Jain, and Rahul Kejriwal. 2021. A large-scale evaluation of neural machine transliteration for indic languages. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 3469–3475, Online. Association for Computational Linguistics.

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

Anoop Kunchukuttan, Pratik Mehta, and Pushpak Bhattacharyya. 2018b. The IIT Bombay English-Hindi parallel corpus. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan. European Language Resources Association (ELRA).

Anoop Kunchukuttan, Ratish Puduppully, and Pushpak Bhattacharyya. 2015. Brahmi-net: A transliteration and script conversion system for languages of the indian subcontinent. In *Proceedings of the 2015 conference of the North American chapter of the association for computational linguistics: demonstrations*, pages 81–85.

Yash Madhani, Mitesh M. Khapra, and Anoop Kunchukuttan. 2023. Bhasa-abhijnaanam: Native-script and romanized language identification for 22 Indic languages. In *Proceedings of the 61st Annual Meeting of the Association for Computational*

Linguistics (Volume 2: Short Papers), pages 816–826, Toronto, Canada. Association for Computational Linguistics.

Arnav Mhaske, Harshit Kedia, Sumanth Doddapaneni, Mitesh M. Khapra, Pratyush Kumar, Rudra Murthy, and Anoop Kunchukuttan. 2022. Naamapadam: A large-scale named entity annotated data for indic languages. *arXiv preprint arXiv:22212.10168*.

Benjamin Muller, Antonios Anastasopoulos, Benoît Sagot, and Djamé Seddah. 2021. When being unseen from mBERT is just the beginning: Handling new languages with multilingual language models. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 448–462, Online. Association for Computational Linguistics.

V. Rudra Murthy and Pushpak Bhattacharyya. 2016. A deep learning solution to named entity recognition. In *Computational Linguistics and Intelligent Text Processing - 17th International Conference, CICLing 2016, Konya, Turkey, April 3-9, 2016, Revised Selected Papers, Part I*, volume 9623 of *Lecture Notes in Computer Science*, pages 427–438. Springer.

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.

Franz Josef Och and Hermann Ney. 2003. A systematic comparison of various statistical alignment models. *Comput. Linguistics*, 29(1):19–51.

Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli. 2019. fairseq: A fast, extensible toolkit for sequence modeling. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Demonstrations*, pages 48–53. Association for Computational Linguistics.

Barbara Plank, Anders Søgaard, and Yoav Goldberg. 2016. Multilingual part-of-speech tagging with bidirectional long short-term memory models and auxiliary loss. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, ACL 2016, August 7-12, 2016, Berlin, Germany, Volume 2: Short Papers*. The Association for Computer Linguistics.

Bedapudi Praneeth. 2020. Anuvaad. `https://github.com/notAI-tech/Anuvaad.git`.

Alec Radford, Jong Wook Kim, Tao Xu, Greg Brockman, Christine McLeavey, and Ilya Sutskever. 2022. Robust speech recognition via large-scale weak supervision. *OpenAI Blog*.

Gowtham Ramesh, Sumanth Doddapaneni, Aravinth Bheemaraj, Mayank Jobanputra, Raghavan AK, Ajitesh Sharma, Sujit Sahoo, Harshita Diddee, Divyanshu Kakwani, Navneet Kumar, et al. 2022. Samanantar: The largest publicly available parallel corpora collection for 11 indic languages. *Transactions of the Association for Computational Linguistics*, 10:145–162.

John Richardson, Toshiaki Nakazawa, and Sadao Kurohashi. 2013. Robust transliteration mining from comparable corpora with bilingual topic models. In *Proceedings of the Sixth International Joint Conference on Natural Language Processing*, pages 261–269, Nagoya, Japan. Asian Federation of Natural Language Processing.

Brian Roark, Lawrence Wolf-Sonkin, Christo Kirov, Sabrina J. Mielke, Cibu Johny, Isin Demirsahin, and Keith B. Hall. 2020. Processing South Asian Languages Written in the Latin Script: the Dakshina Dataset. In *Proceedings of The 12th Language Resources and Evaluation Conference, LREC 2020, Marseille, France, May 11-16, 2020*, pages 2413–2423. European Language Resources Association.

Rishiraj Saha Roy, Monojit Choudhury, Prasenjit Majumder, and Komal Agarwal. 2013. Overview of the fire 2013 track on transliterated search. In *Post-Proceedings of the 4th and 5th Workshops of the Forum for Information Retrieval Evaluation*, FIRE '12 & '13, New York, NY, USA. Association for Computing Machinery.

Hassan Sajjad, Alexander Fraser, and Helmut Schmid. 2012. A statistical model for unsupervised and semi-supervised transliteration mining. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 469–477.

Vishnu Sukthankar. 2017. The Mahabharata. `https://sanskritdocuments.org/mirrors/mahabharata/mahabharata-bori.html`.

Raghavendra Udupa, Saravanan K, Anton Bakalov, and Abhijit Bhole. 2009. "They Are Out There, If You Know Where to Look": Mining transliterations of oov query terms for cross-language information retrieval. In *Advances in Information Retrieval*, pages 437–448, Berlin, Heidelberg. Springer Berlin Heidelberg.

Raghavendra Udupa, K Saravanan, A Kumaran, and Jagadeesh Jagarlamudi. 2008. Mining named entity transliteration equivalents from comparable corpora. In *Proceedings of the 17th ACM conference on Information and knowledge management*, pages 1423–1424.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 5998–6008.

Denny Vrandečić and Markus Krötzsch. 2014. Wikidata: A free collaborative knowledgebase. *Commun. ACM*, 57(10):78–85.

A Existing Sources of Transliteration corpora

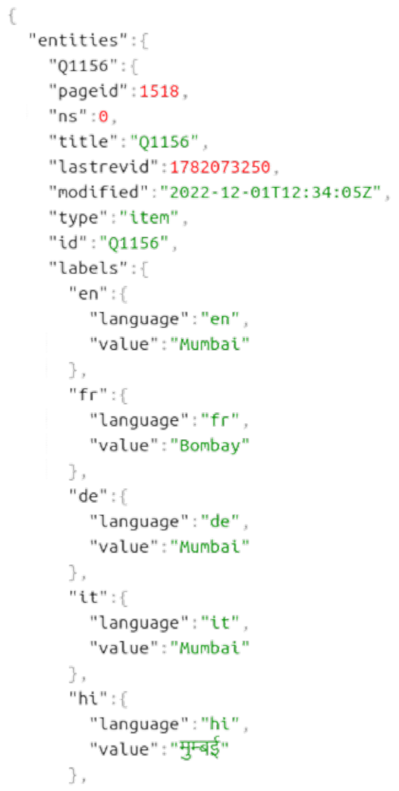
We compiled several existing sources, with the majority of the data coming from the Dakshina (Roark et al., 2020) and the Brahminet (Kunchukuttan et al., 2015) corpora. In addition, we also compiled other small datasets, including Xlit-Crowd (Khapra et al., 2014), Xlit-IITB-Par (Kunchukuttan et al., 2018b), FIRE 2013 Track on Transliterated Search (Roy et al., 2013), NotAI-tech-English-Telugu (Praneeth, 2020), and AI4Bharat StoryWeaver Xlit Dataset (Benjamin and Gokul, 2020). Table 10 provides statistics on the transliteration corpora compiled from existing sources.

B Example of Wikidata mining

Figure 1 describes the structure of wikidata database. As shown in Figure 1, Each entity has *labels* that are common names of the items in multiple languages. We have the location entity *Mumbai* with its translitertions in multiple Indian languages. We extract such English-Indian language label pairs creating transliteration pairs. For multi-word labels, we use a simple method that worked well: creating all possible word pair candidates from the English and the Indian language labels followed by filtering the candidate pairs using the automatic transliteration validator described in Section 4.3. For example, the multi-word pair "Mahatma Gandhi" will result in 4 candidate pairs { Mahatma महात्मा, Mahatma गांधी, Gandhi महात्मा, Gandhi गांधी } We then filter these candidate pairs using the automatic transliteration validator described in section 4.3. It will filter out these two incorrect pairs, { Mahatma महात्मा } and { Gandhi गांधी }.

C Details of Mining from Translation Corpora

Table 11 describes the examples from Samanantar parallel translation corpora. Transliteration pairs



```
{
  "entities":{
    "Q1156":{
    "pageid":1518,
    "ns":0,
    "title":"Q1156",
    "lastrevid":1782073250,
    "modified":"2022-12-01T12:34:05Z",
    "type":"item",
    "id":"Q1156",
    "labels":{
      "en":{
        "language":"en",
        "value":"Mumbai"
      },
      "fr":{
        "language":"fr",
        "value":"Bombay"
      },
      "de":{
        "language":"de",
        "value":"Mumbai"
      },
      "it":{
        "language":"it",
        "value":"Mumbai"
      },
      "hi":{
        "language":"hi",
        "value":"मुम्बई"
      },
```

Figure 1: Structure of Wikidata record. labels in multiple languages attribute to transliterations pairs

are highlighted in these examples. As described in the section 3.2, these transliteration pairs could be in the form of named entities, loan words, and cognates in parallel translation sentences.

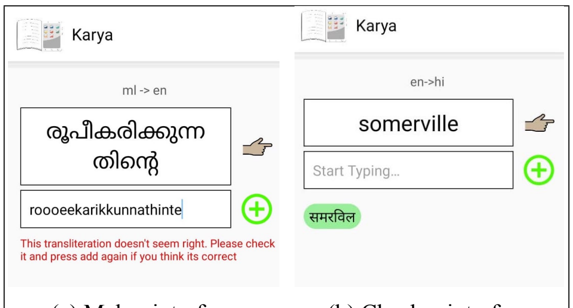
Unsupervised method by Sajjad et al. (2012) The unsupervised method suggested by Sajjad et al. (2012) is implemented in the transliteration module (Durrani et al., 2014a) of *Moses* (Koehn et al., 2007), to mine transliteration pairs from the word alignments by distinguishing transliterations and non-transliterations. Their model structure is motivated as follows: A combination of a transliteration sub-model and a non-transliteration model, combined with interpolation weights. The parameters of t he transliteration model and the interpolation weights are learned during the training whereas the parameters of the non-transliteration sub-model are kept fixed after initialization. Their training procedure ensures that the transliteration model assigns most of the probability mass to transliteration pairs, whereas the non-transliteration sub-model evenly distributes the probability mass across all possible source and target word pairs. Hence, the trained model assigns a higher score to the transliteration pairs and thus helps in identifying such pairs.

	ben	guj	hin	kan	kok	mai	mal	mar	pan	snd	sin	tam	tel	urd
Dakshina	95	105	44	51	-	-	58	56	71	39	42	68	59	106
Xlit-Crowd	-	-	11	-	-	-	-	-	-	-	-	-	-	-
Xlit-IITB-Par	-	-	69	-	-	-	-	-	-	-	-	-	-	-
FIRE-2013-Track	5	1	36	-	-	-	-	-	-	-	-	-	-	-
AI4B-StoryWeaver	-	-	101	-	60	103	-	-	-	-	-	-	-	-
NotAI-tech En-Te	-	-	-	-	-	-	-	-	-	-	-	-	39	-
Brahminet	8	7	11	-	6	-	3	5	9	-	-	4	5	6
Total unique word pairs	104	111	234	51	65	102	61	60	78	39	42	71	97	111

Table 10: Statistics of transliteration pairs compiled from existing sources. All numbers are in thousands.

eng	**hin**
From the Azad Kashmir Regiment, Lt Gen Afgun has commanded a Division on the LOC when Gen Bajwa was commander of the X Corps	आजाद कश्मीर रेजिमेंट से लेफ्टिनेंट अफगुन ने एलओसी पर एक डिविजन कमांड किया है, जब जनरल बाजवा टेंथ कॉर्प्स के कमांडर थे

Table 11: Examples of transliteration pairs from the Samanantar parallel translation corpus.



This transliteration doesn't seem right. Please check it and press add again if you think its correct

(a) Maker interface (b) Checker interface

Figure 2: Annotation UI in the *Karya* app.

Please refer to Sajjad et al. (2012) for more details.

D Annotation Process in detail

Karya App We use *Project Karya* (Chopra et al., 2019; Abraham et al., 2020), an open-source crowdsourcing platform making digital language work more inclusive and accessible to the masses using smartphones, as our annotation platform. The app is used for collecting transliteration data from selected annotators. The user interface is shown in Figure 2.

Annotators detail In all, we employ 68 annotators from two data annotation agencies as *transliterators* and *validators*, with the latter having more experience in linguistic tasks. The annotators were paid INR 2 (USD 0.026) per native language word.

Transliteration annotation task Each transliteration micro-task contains 100 native words to be transliterated and then validated post transliteration. Transliterators are instructed to write transliterations that are natural. A rule-based automatic transliteration validator (Appendix 4.3) is used as the first level of internal quality check for the proposed transliterations. The validator can reject wrong variants and enter new variants for a native script word missed by the transliterator on a similar interface, as shown in Figure 2b. The variants accepted or added by the validator constitute the final set of romanized variants for the input word.

b	ಬ	ಭ					**p**	ಪ	ಫ
c	ಕ	ಚ	ಛ	ಶ	ಷ	ಸ	**q**	ಕ	

Table 12: Snippet of Kannada consonant mapping.

E Automatic Validation Algorithm

The transliteration checker is based on the Transliteration Equivalence algorithm for English (Roman script)-Hindi described in Khapra et al. (2014) which checks equivalence of the consonant mappings in a potential transliteration pair. To achieve this, the algorithm takes two pieces of information: (i) a stop-list of vowels in the two languages, and (ii) a list of consonant mappings between the two languages. We incorporate these rules and extend the above-mentioned approach to other Indic languages with the aid of language experts. Table 12 shows a snippet of consonant mappings for the Kannada language. There is a large overlap in the consonant mapping rules across Indian languages, but we also cater to language-specific exceptions. The transliteration validator firstly removes vowels and all characters present in a stop-list from the English variant and maps each English consonant to the relevant Indic language consonant according to the consonant mapping table as shown in Table 12. Once all possible Indic language variants of the English word are formulated, they are compared against the original Indic word to check validity of the romanized transliteration. We check the effectiveness of the transliteration validator on transliteration pairs in the Dakshina train set and observe that it achieves a minimum accuracy of 90% across languages as shown in Table 15. This indicates its utility and non-intrusiveness.

F IndicXlit: Model parameters and Training details

Vocabulary The input vocabulary is the set of Roman script characters found in the training set, while the output vocabulary is the union of characters from various Indic language scripts found in the training set. The input and output vocabulary sizes are 28 and 780 characters, respectively.

Model parameters We experimented with different hyperparameters for the model architecture and following parameters gave the best results on the Dakshina development set. The IndicXlit model has 6 encoder and decoder layers each, 256 dimensional input embeddings, feedforward network (FFN) dimension of 1024, and 4 attention heads. We use GELU activation function (Hendrycks and Gimpel, 2016) in the feedforward layer, and dropout=0.5. We preprocess multi-head attention, encoder attention, and each layer of FFN with layernorm. We add layer normalization to the embeddings (Ba et al., 2016).

Training Details We use Fairseq (Ott et al., 2019) for training our transliteration models, specifically the *translation multi simple epoch task*. We optimize the cross-entropy loss using the Adam optimizer (Kingma and Ba, 2015) with Adam-betas of (0.9, 0.98). We use a peak learning rate of 0.001, 4000 warmup steps, and the *inverse-sqrt* learning rate scheduler. We use a global batch size of 4096 pairs. Each minibatch contains examples from all language pairs. Due to the skew in data distribution across languages, we use temperature sampling (Arivazhagan et al., 2019) to oversample data from low-resource languages with temperature $T = 1.5$. We optimize the above mentioned values of the hyperparameters over the Dakshina training and development set. We train the model on 4 A100 GPUs for a maximum of 50 epochs. Table 5 shows the statistics of the train and validation splits used to train IndicXlit.

G Top-3, Top-5 and F1-scores

Table 13 describes the Top-3, Top-5 accuracies and F1-scores of IndicXlit model and models trained on Dakshina train set on Dakshina test set. Table 14 describes the Top-3, Top-5 accuracies and F1-scores of IndicXlit model on Aksharantar test set.

H Error Analysis in detail

To understand the errors made by IndicXlit, we analysed the output of the model for 100 randomly sampled words each for *ben, guj, hin, kan, mar, pan, tel* from the Dakshina dataset. Table 16 summarizes the major transliteration errors.

Vowels. The most common errors across languages are with respect to vowels, as reported in previous studies (Kunchukuttan et al., 2021). Insertion/deletion of the '◌ा ' vowel diacritic along with confusion between short/long vowel diacritics constitute a large fraction of transliteration errors.

Similar consonants. Another common source of errors is confusion between similar consonants, as shown in Table 16.

Gemination. Other prominent errors are with respect to gemination (*e.g.,* {input: *thathvavethaga*, reference: తత్వవేత్తగా, prediction: తత్వవేతగా}).

Acronyms. Acronyms have a peculiar transliteration behavior that needs to be handled differently (*e.g.,* {input: *wsd*, reference: डब्ल्यूएसडी *(wsd)*, prediction: वास्ड}(wasd)).

Contextual ambiguity. The "other errors" category is the result of ambiguities that cannot be easily resolved from character context. These are prevalent across all the testsets to varying degrees.

Language-specific. In addition, we observed some language-specific error categories. For example, in Gujarati, there is ambiguity between '◌ં'(anusvara) and 'ન'(n) characters; in Marathi, there are instances of deletion of ''◌्'' diacritic; in Punjabi, there are instances of addition/deletion of 'ੳ', '◌ੱ' and '◌ੰ' vowels/diacritics; in Bengali, there is ambiguity between 'শ'(sha) and 'স'(sa); in Kannada, confusion exists between consonants 'ಶ'(sha) and 'ಸ'(sa), as well as 'ಳ'(lla) and 'ಲ'(la);

Model	ben	guj	hin	kan	mal	mar	pan	snd	sin	tam	tel	urd	avg
Roark et al. (2020)	49.4	49.5	50.0	66.2	58.3	49.7	40.9	33.2	54.7	65.7	67.6	36.7	51.83
Our models trained on Dakshina dataset, Top-1 accuracy													
Bilingual	41.8	42.7	46.7	58.3	52.8	41.4	37.3	35.0	52.4	56.0	63.2	34.7	46.9
Multilingual	47.2	51.0	51.8	66.4	56.5	51.0	42.2	41.3	58.7	63.5	67.1	38.3	52.9
Our models trained on Dakshina dataset, Top-3 accuracy													
Bilingual	64.3	68.0	71.4	80.0	71.8	64.8	62.4	59.1	76.8	75.0	81.4	59.3	69.5
Multilingual	68.8	75.1	74.6	85.2	75.4	74.8	68.0	66.1	81.8	79.8	85.0	62.0	74.7
Our models trained on Dakshina dataset, Top-5 accuracy													
Bilingual	72.0	76.1	78.1	86.4	77.5	72.7	71.6	68.1	83.3	79.2	85.8	68.0	76.6
Multilingual	75.9	82.0	81.5	89.8	80.6	80.8	76.7	74.6	87.4	83.2	89.3	71.1	81.1
Our models trained on Dakshina dataset, F1-score accuracy													
Bilingual	89.8	91.4	90.7	94.7	92.6	90.7	88.1	86.5	92.8	92.8	95.0	87.1	91.0
Multilingual	91.2	92.8	91.8	95.7	93.3	92.4	89.2	88.1	93.7	93.7	95.5	87.9	92.1
IndicXlit model													
Top-1	**55.4**	**62.0**	**60.5**	**77.1**	**63.5**	**64.8**	**47.2**	**48.5**	**63.9**	**68.1**	**73.3**	**42.1**	**60.5**
Top-3	75.7	82.9	81.9	89.9	79.5	81.6	72.3	70.9	85.5	82.0	88.4	64.9	79.6
Top-5	81.6	88.5	87.1	92.9	83.5	86.3	80.4	78.9	90.1	85.9	91.3	72.5	84.9
F1-score	92.5	94.4	93.4	97.0	94.0	94.2	90.1	89.4	94.4	94.3	96.3	88.8	93.2

Table 13: Comparing Top-3, Top-5 accuracies and F1-score reported on the Dakshina test set.

similarly in Telugu, there is ambiguity between 'ళ'(lla) and 'ల'(la).

Valid alternatives. Finally, some of the reported transliteration errors are actually valid alternative transliterations (*e.g.,* {input: *khurasan*, reference: खुरासान, prediction: खुरासन}).

Testset	asm	ben	brx	kok	guj	hin	kan	kas	mai	mal	mni	mar	nep	ori	pan	san	tam	tel	urd	avg
Top-3 accuracy																				
Dakshina	-	75.7	-	-	82.9	81.9	89.9	-	-	79.5	-	81.6	-	-	72.3	-	82.0	88.4	64.9	79.9
AK-Freq	83.8	84.7	89.1	84.4	87.6	79.1	91.4	52.9	93.2	86.1	94.2	88.7	92.7	83.6	69.5	96.2	88.2	96.9	-	85.7
AK-Uni	76.0	76.3	80.9	77.5	75.9	73.0	85.8	46.4	81.6	74.3	-	68.8	89.8	70.3	50.4	91.0	80.8	90.4	68.2	75.4
AK-NEF	61.2	54.1	48.7	46.7	66.7	75.7	72.1	22.6	58.8	46.9	64.0	70.6	71.2	43.5	54.0	41.4	61.8	71.8	74.8	58.3
AK-NEI	63.0	62.9	49.1	62.4	71.7	81.7	71.6	40.9	72.8	56.3	68.9	77.4	77.4	52.1	61.2	51.4	66.4	72.0	71.4	64.8
Micro-avg	73.6	73.9	68.3	70.8	81.1	79.1	86.0	44.3	80.0	75.2	82.6	79.1	83.0	63.1	67.2	74.8	79.6	86.5	66.4	74.5
Top-5 accuracy																				
Dakshina	-	81.6	-	-	88.5	87.1	92.9	-	-	83.5	-	86.3	-	-	80.4	-	85.9	91.3	72.5	85.0
AK-Freq	88.3	89.6	91.4	88.8	92.6	84.6	94.3	62.9	96.1	90.2	95.7	92.2	95.7	88.4	74.8	97.8	90.1	97.2	-	89.5
AK-Uni	82.4	80.9	85.6	82.1	82.9	80.1	90.2	54.6	86.9	78.7	-	74.2	91.6	75.8	57.2	94.3	84.9	93.1	74.7	80.6
AK-NEF	67.7	61.3	57.6	52.4	72.9	83.1	78.0	29.5	67.8	52.6	71.1	77.0	78.2	50.8	61.8	49.6	70.6	79.5	80.9	65.4
AK-NEI	71.1	68.6	57.4	67.8	79.1	87.0	78.4	47.0	79.0	63.6	75.7	83.0	83.3	59.5	68.4	59.5	73.7	79.7	77.6	71.5
Micro-avg	79.9	79.8	74.1	75.8	86.9	84.9	89.9	52.9	85.3	79.8	86.5	84.0	87.4	69.4	74.8	79.5	84.0	89.9	73.6	79.9
F1-Score																				
Dakshina	-	92.5	-	-	94.4	93.4	97.0	-	-	94.0	-	94.2	-	-	90.1	-	94.3	96.3	88.8	93.5
AK-Freq	94.9	94.2	96.4	94.4	95.2	94.0	97.4	86.0	96.1	96.9	97.6	96.7	96.9	94.5	90.2	97.7	97.4	98.6	-	95.3
AK-Uni	94.7	94.6	96.2	95.5	95.2	94.1	97.7	85.8	95.3	96.0	-	94.6	98.1	94.2	89.8	97.6	97.0	97.9	91.9	94.8
AK-NEF	89.6	87.6	88.2	85.4	91.3	92.9	93.3	80.6	87.8	85.5	91.0	91.7	91.3	87.5	87.3	85.2	91.5	93.3	91.5	89.1
AK-NEI	90.7	90.4	87.7	89.4	92.8	94.3	92.8	85.4	91.3	87.7	91.6	93.4	92.7	88.8	89.7	86.9	92.3	92.8	91.7	90.7
Micro-avg	93.2	92.4	92.4	92.1	94.4	93.7	96.5	85.2	93.6	93.8	94.9	94.5	94.9	91.4	89.9	93.0	94.6	96.1	89.7	93.0

Table 14: Top-3, Top-5 accuracy and F1-score for IndicXlit on various testsets.

ben	guj	hin	kan	mal	mar	pan	tam	tel
89.9	97.4	94.9	97.9	84.5	97.7	97.2	92.9	96.0

Table 15: Accuracy of Automatic Transliteration Validator on Dakshina dataset.

Types of errors	%	Most common errors across all languages
Vowel errors	45	Vowels are getting interchanged, model is skipping or adding '◌ा'
Interchanging short, long vowels	15	*{*'◌े'⇌ '◌ै'*}*, *{*'◌ि'⇌ '◌ी'*}*, *{*'◌ृ' ⇌ '◌ु'*}*, *{*'◌ॉ' ⇌ '◌ो'*}*, *{*'◌ौ' ⇌ '◌ो'*}*, *{*'◌े' ⇌ '◌े'*}*, *{*'आ' ⇌ 'अ'*}*
Consonant errors	25	*{*'ड' ⇌ 'द'*}*, *{*'त' ⇌ 'ट'*}*, *{*'ण' ⇌ 'न'*}*
Other errors	15	Acronyms, gemination errors, silent characters, *valid* alternative transliterations, unnecessary vowel suppressor addition

Table 16: Summary of errors of IndicXlit outputs.