Assamese Back Transliteration - An Empirical Study Over Canonical and Non-canonical Datasets

Hemanta Baruah[1], **Sanasam Ranbir Singh**[1,2], **Priyankoo Sarmah**[1,3]
[1]Centre for Linguistic Science and Technology
[2]Department of Computer Science and Engineering
[3]Department of Humanities and Social Sciences
Indian Institute of Technology, Guwahati, Assam, India
{hemanta.b[1], ranbir[1,2], priyankoo[1,3]}@iitg.ac.in

Abstract

This study evaluates the performance of transformer-based state-of-the-art machine transliteration systems in processing noisy transliterated texts (non-canonical form) from social media in the context of the Assamese language. Fifteen different setups, using both canonical (manually forward transliterated) and non-canonical (manually back transliterated) datasets, are examined to identify suitable combinations for handling both the types of data. The findings indicate that existing transliteration models trained on manually transliterated Romanized datasets fail to handle highly ambiguous and noisy characteristics of social media data, resulting in degraded performance. This study systematically investigates how to combine noisy social media and manually transliterated clean datasets to build an effective transliteration model suitable for handling both clean and noisy text. To the best of our knowledge, this paper represents the pioneering effort in the domain of social media for the Assamese language machine transliteration.

1 Introduction

With increase in the availability of user generated text in phonetically transliterated form in digital platforms, specially on social media platforms, *machine transliteration*, the task of phonetic translation of text from one orthography to another orthography is becoming crucial for various natural language processing (NLP) applications. The situation become more critical for the multilingual society like India. A significant amount of Indian language contents on social media are written using *Roman script*. Earlier studies on machine transliteration were mostly explored as a sub-task of machine translation for dealing with name entities or technical terms (Banchs et al., 2015). As the presence of phonetically transliterated text is becoming ubiquitous today for regular communication on various social media platforms, developing a machine transliteration system is an important research problem. Further, depending on the nature of the orthographies used in source and target text, machine transliteration may be of two types -transliteration, and back transliteration. Transliteration generally refers to converting a text in a language written using native script of the language into another phonetically same/close text with another script (Vyshnavi et al., 2022). Back transliteration refers to converting a transliterated text in a language written using non-native script back to corresponding text in native script (Knight and Graehl, 1997).

From the nature of the dataset used, we may broadly group machine transliteration studies into two - *canonical parallel dataset* and *non-canonical parallel dataset*. In a canonical parallel dataset, a legitimate text in a language in its native script is transliterated to another orthography, mostly by the language experts. Such datasets have no or limited variations. Whereas, non-canonical forms refer to noisy transliterated text available on unrestricted platforms like social media platforms. User may generate the transliterated text without following any rules or norms. In the case of a non-canonical parallel dataset, such noisy transliterated text is back transliterated to its native script by language experts.

The aim of this study is to identify suitable combinations that effectively handle both the clean and noisy transliterated text. We systematically examine fifteen different setups to address transliteration challenges, leveraging both canonical (manually forward transliterated native text) and non-canonical (manually back transliterated noisy social media) dataset. Through this empirical investigation, we investigate the limitations of existing transliteration models when applied to highly ambiguous and noisy social media data. Our findings reveal that transliteration models trained solely on manually transliterated canonical Ro-

manized dataset exhibit limited performance when confronted with non-canonical noisy social media text. Similarly, models trained on non-canonical Romanized social media dataset struggle to perform well on canonical Romanized test set. Then we tried with a merge setup combining both canonical and non-canonical dataset together. We have noticed that the merge setup somehow able to handle both the types of data to some extent. Finally we tried with a transfer learning setup. Transfer learning setup also failed to perform well on both the types of data. These observations underscore the need for developing robust transliteration models specifically designed to handle the complexities of both clean and noisy transliterations in diverse linguistic contexts. This research is motivated by the growing importance of harnessing the vast amount of social media data for real-time information processing and the need to accurately transliterate user-generated content in multilingual countries like India. The study focuses on Assamese, an Eastern Indo-Aryan language, which holds a prominent position among India's 22 scheduled languages, serving as the first language for nearly 15.3 million native speakers (Chandramouli and General, 2011) and as the official language of Assam, a northeastern state in the country.

In the subsequent sections, we conduct a comprehensive investigation of our dataset, different experimental setups, performance evaluations, and error analysis. Our primary goal is to explore and identify the most effective strategies for building a robust transliteration model that can proficiently handle both canonical and non-canonical transliterated texts in Assamese language. Throughout the paper, we will refer to the setup using the non-canonical parallel dataset as the "noisy model" and the setup using the canonical parallel dataset as the "clean model" respectively. Additionally, the models trained with a combination of both non-canonical and canonical parallel dataset will be referred to as the "merge model". Through this careful examination, we aim to gain valuable insights and develop an efficient transliteration system tailored to the intricacies associated with processing Assamese text.

2 Dataset

To obtain the required non-canonical Romanized Assamese dataset, we chose three prominent social media platforms: Facebook, YouTube, and Twitter. We carefully curated specific sources from each platform that host abundant collections of Romanized Assamese datasets. For YouTube, we used the YouTube Data API[1] to collect comments from selected Assamese YouTube channels[2]. On Twitter, our focus was on gathering mainly the reply tweets from a popular Assamese Twitter handle[3], utilizing the Tweepy API[4]. Similarly, we extracted comments from selected Facebook pages[5] using the Facebook Graph API[6]. Details of the complete experimental dataset and data collection duration are provided in Table 1. After collecting the data, we locally developed a transliteration tagging tool and engaged 24 online annotators. The transliterators were ensured to be fluent in both Assamese and English writing. While transcribing, the embedded non-native words were left in their original forms. Social media noise, such as spelling errors, shortened and elongated forms of Romanized native words, were normalized and transliterated back to their correct Assamese orthographic forms.

To obtain the canonical Assamese dataset, we made use of the publicly available Aksharantar dataset Madhani et al. (2022). This dataset comprises Romanized Assamese to native Assamese transliteration pairs, which were generated using a combination of strategies. These strategies encompassed extracting transliteration pairs from parallel corpora, mining from monolingual corpora, utilizing existing transliterated data sources, and carefully constructing a transliteration corpus through manual forward transliteration from native Assamese words to their corresponding Romanized forms. In total, the Aksharantar dataset provided us with 1,87,924 Assamese transliteration pairs for analysis and evaluation. We performed further data cleaning and observed some errors that needed attention. Specifically, we identified a total of 8905 instances where the Bengali alphabet "র" was mistakenly used instead of the correct Assamese alphabet "ৰ". Conversely, the Assamese alphabet "ৰ" appeared correctly 1,33,367 times in the entire corpus. Unlike some other Indian languages, the diacritic **Nukta** (" ়") is not treated as

[1] YouTube Data API
[2] News Live, Assamese Mixture, Dimpu's Vlogs
[3] https://twitter.com/himantabiswa
[4] https://docs.tweepy.org/en/stable/
[5] GU Confession Page, CMO Assam Page
[6] Facebook Graph API

Table 1: Statistics of the collected dataset from three major social media sources along with the duration

Social Media Sources	Duration of Data Collection	#posts collected	#posts annotated	#words (total)	#unique Assamese words (total)
Facebook	Dec-2013 to Feb-2017	4,09,168	5,300	71,800	79,200
YouTube	Jun-2018 to Aug-2023	3,85,676	50,000	4,26,089	
Twitter	Mar-2021 to Aug-2021	2,85,676	5,010	91,400	

Table 2: Training, Validation and Testing dataset statistics (All transliteration pairs are word splitted into characters)

	DATASET SIZE			
	TRAINING	**VALIDATION**	**TESTING**	**TOTAL**
Noisy setup	45,934	6,560	13,120	65,614
Clean setup	1,31,546	18,792	37,586	1,87,924
MergeUniqueTrans	1,74,696	19,411	Noisy and Clean test	2,44,813
MergeDuplicateTrans	4,89,266	54,363	Noisy and Clean test	5,94,335
IndicXlitEval	Multilingual IndicXlit Model		Noisy and Clean test	-

a separate character in Assamese, resulting in individual characters like "য়", "ড়", and "ঢ়". These characters are not realized as combinations, such as য + ় = য়, ড + ় = ড় or ঢ + ় = ঢ়. We corrected all such instances to their proper forms as "য়", "ড়", and "ঢ়" respectively.

3 Experimental Setup

For conducting our experiments, we employed a transformer-based encoder-decoder architecture (Vaswani et al., 2017) to create a character-level monolingual transliteration model. This model operates in a one-to-one setting, taking a Romanized character sequence as input and generating the corresponding character sequence in the native Assamese script as output. To implement the monolingual transliteration task, we utilized the Fairseq implementation (Ott et al., 2019) of the transformer architecture. The model architecture included 6 encoder and 6 decoder layers, with layernorm applied within each transformer layer to normalize the outputs of the multi-head self-attention and feed-forward sub-layers. The GELU activation function was used, and the number of attention heads was set to 4 for both the encoder and decoder self-attention layers. We utilized a batch size of 1024 and set the dimensions for the encoder and decoder embeddings to 256 and the dimensions for the encoder and decoder feed-forward networks (FFN) to 1024. Dropout was applied with a rate of 0.5 and attention dropout at 0.1. We used label-smoothed cross-entropy with a smoothing factor 0.1 as the training criterion. The Adam optimizer was employed with betas (0.9, 0.98) and a learning rate of 0.001. The learning rate scheduler was set to inverse square root with a warmup initialization learning rate of 0 and warmup updates of 4000. The model was trained for a maximum of 100 epochs for each setups.

Our experiments were conducted across three main setups: (1) Noisy test set, (2) Clean test set, and (3) Fine-tuning. Within the first two setups, we conducted six and five experiments respectively, while the fine-tuning setup involved four experiments. Thus, we conducted a total of fifteen distinct experimental setups. We employed a 70-10-20 split to ensure comprehensive evaluation for training, validation, and testing purposes. Among these fifteen setups, three specifically incorporated the state-of-the-art multilingual transliteration model, IndicXlit (Madhani et al., 2022), for comparative analysis. Detailed dataset statistics for all fifteen setups can be found in Table 2. In the following sections, we will elaborate on our experimental setups.

1. **Noisy Setup:** In this setup, we specifically focused on the non-canonical transliteration pairs available in our dataset for training and validating our model. Subsequently, the trained model was employed to evaluate the performance on both the noisy and clean test sets. For this setup, the input and output vocabulary sizes are 44 and 76 characters, respectively.

2. **Clean Setup:** Similarly, in this setup, we utilized the canonical transliteration pairs from Aksharantar dataset for training and validating the model. Subsequently, we evaluated the model's performance on both noisy and clean test sets using the same trained model. The input and output vocabulary sizes for this setup are 28 and 68 characters, respectively.

3. **Merge Transliteration Model with Unique Pairs (MergeUniqueTrans):** In this setup, we combined all the unique transliteration

Table 3: Transliteration result in terms of Accuracy@1, Accuracy@4, Character Accuracy, Mean F-score, MRR and BLEU4 score for all the setups (Experimental setups with the highest and lowest accuracies(top-1) are highlighted in red and blue, respectively, while the accuracy of the best-performing setup is highlighted in **bold**)

			Acc@1	Acc@4	Char Accuracy	Mean F-score	MRR	BLEU4 Score
Setup with Noisy Test Set	**Noisy Model (noisy training set unique)**	***setup 1***	41.57	69.79	79.85	87.57	53.38	69.15
	Clean Model (clean training set unique)	***setup 2***	28.21	42.76	73.76	83.12	34.05	56.64
	Merge Model (with unique pairs)	***setup 3***	**41.88**	69.20	81.71	87.74	53.29	69.97
	Merge Model (with duplicate pairs)	***setup 4***	46.45	74.08	82.65	88.74	57.94	73.78
	IndicXlit Model (multilingual model)	***setup 5***	26.87	42.46	71.57	82.18	33.19	54.03
	Noisy Model (noisy training set with duplicates)	***setup 6***	46.22	73.71	82.66	88.61	57.69	73.88
Setup with Clean Test Set	**Clean Model (clean training set)**	***setup 7***	77.78	92.99	96.59	97.36	84.58	91.94
	Noisy Model (noisy training set)	***setup 8***	34.71	56.36	85.71	89.26	43.74	65.29
	Merge Model (with unique pairs)	***setup 9***	**75.04**	91.91	95.82	96.77	82.56	90.85
	Merge Model (with duplicate pairs)	***setup 10***	67.47	86.17	93.69	95.33	75.59	87.22
	IndicXlit Model (multilingual model)	***setup 11***	76.85	92.09	96.38	97.27	83.68	90.93
Finetunning Setup	**Fine-tune Clean Model (on noisy dataset)**	***setup 12***	36.63	62.11	77.43	85.66	47.24	64.40
	Fine-tune IndicXlit Model (on noisy dataset)	***setup 13***	38.05	62.42	77.26	86.19	48.16	66.48
	Fine-tune Noisy Model (on clean dataset)	***setup 14***	77.07	92.87	96.46	97.25	84.13	91.56
	Evaluate *setup 14* (on noisy testset)	***setup 15***	28.16	43.29	74.54	83.6	34.21	56.88

pairs from both the noisy and clean datasets for training and validation. Following training, the same trained model can be employed to assess performance using both the noisy and clean test sets.

4. **Merge Transliteration Model with Duplicate Pairs (MergeDuplicateTrans):** Similarly, in this setup, we combined all available datasets, including noisy pairs from the Noisy model setup and clean pairs from the Clean model setup, while also including duplicate transliteration pairs this time. We aim to prioritize frequently occurring transliteration pairs, thus normalizing the dataset and assigning weights to high-frequency pairs. Subsequently, the trained model was used to assess the performance on both clean and noisy test sets.

5. **IndicXlit Model (IndicXlitEval):** In this setup, we evaluated the performance of the state-of-the-art multilingual IndicXlit model using the test sets from both the noisy and clean models. The input and output vocabulary sizes for this multilingual setup are 28 and 780 characters, respectively.

6. **Finetune (noisy model):** In this setup, we fine-tuned our noisy model, which was initially trained on the noisy dataset, using the clean dataset of the clean model and evaluated its performance on both noisy and clean test set.

7. **Finetune (clean model):** Likewise, here we fine-tuned our clean model, which was initially trained on the clean dataset, using the dataset from the noisy model, which contains the noisy dataset.

8. **Finetune (IndicXlit model):** In this setup, we fine-tuned the multilingual IndicXlit model using our noisy model dataset and subsequently evaluated its performance.

4 Result and Discussion

In this section, we discuss the performance of various transliteration setups using multiple evaluation metrics, including top-1 Accuracy (Acc@1),

top-4 Accuracy (Acc@4), Char Accuracy, Mean F-score, MRR score and BLEU4(upto 4 grams) score. The results of all fifteen experimental setups are presented in Table 3. Our analysis reveals that setup 7 achieves the highest accuracy (Top 1) of 77.78%, as expected, where both clean datasets are used for training and testing. On the other hand, setup 5 exhibits the lowest accuracy (Top 1) of 26.87%, where we evaluated the performance of the multingual IndicXlit model on noisy test set. Additionally, we observe that Acc@4 (Top 4) consistently outperforms Acc@1(Top 1) across all the setups. Below we will give a detailed discussion on each of the observations.

1. **Performance Evaluation of Noisy and Clean Transliteration Setups:** Based on our observations, the model trained on both noisy and clean datasets and tested with the respective noisy and clean test sets in setup 1 and setup 7 achieved accuracies of 41.57% and 77.78%, respectively. However, these same models did not perform well when tested with the clean test set in setup 8, resulting in an accuracy of 34.71% for the model trained on noisy data, and an accuracy of 28.21% for the model trained on clean data and tested with the noisy test set in setup 2.

2. **Performance Evaluation on Setups With Noisy and Clean Test Set:** The setup that performed the best on the noisy test set was the Merge Model (with duplicate pairs) in setup 4, achieving the highest accuracy of 46.45%. On the other hand, for the clean test set, the clean model trained on the clean dataset in setup 7 achieved the highest accuracy of 77.78% among all the setups.

3. **Impact of Combining Noisy and Clean Datasets with Unique Pairs on Transliteration Model:** In setup 3, we observed a slight increase in accuracy, from 41.57% (setup 1) to 41.88%, by combining both the noisy and clean training sets using unique pairs and testing with the noisy test set. However, when evaluating the same setup with a clean test set in setup 9, the accuracy decreased to 75.04%, which is slightly lower than the 77.78% achieved in setup 7. This suggests that although the transliteration model did not show a significant improvement from being trained with a combined (unique) noisy and clean dataset, it stands out as the single setup that can handle both the noisy and clean dataset together, compared to all other setups.

4. **Effect of Combining Noisy and Clean Datasets with Duplicate Pairs on Transliteration Model:** Combining the noisy and clean datasets with duplicate pairs in setup 4 and setup 10 yielded inconclusive results. In setup 4, the performance was the highest among all the setups tested on the noisy test set, reaching 46.45% accuracy. However, the same setup achieved only 67.47% accuracy when tested with the clean test set, which is lower than the 77.78% and 75.04% accuracy achieved in setup 7 and setup 9, respectively. One plausible reason for this discrepancy could be the variation in the number of duplicate pairs between our noisy and clean datasets. The higher number of duplicate pairs in the noisy dataset, compared to the clean dataset, might be contributing to the observed variations in performance between these two setups.

   We conducted additional investigations to determine whether merging with duplicate pairs, which includes both the clean and noisy datasets and tested on the noisy test set (setup 4), provides any advantage over using a single noisy model trained with duplicate noisy pairs in setup 6 or not. Interestingly, we observed that setup 6 achieved a comparable accuracy of 46.22% in contrast to the 46.45% accuracy achieved in setup 4. Therefore, our conclusion is that merging with duplicates can be beneficial, but it requires a balanced proportion of both the clean and noisy datasets. Otherwise, a single model trained with duplicate pairs can yield similar results.

5. **Performance of Multilingual IndicXlit Model:** The state-of-the-art multilingual IndicXlit model, mainly trained on clean or canonical datasets, exhibits suboptimal performance on the noisy test set (setup 5) with an accuracy of only 26.87%, compared to 41.57% in setup 1. However, when evaluated with a clean test set, the same model demonstrates a comparable accuracy of 76.85% (setup 11), similar to the 77.78% accuracy achieved in setup 7. The slight difference in performance between setup 11 and setup 7

may be attributed to out-of-vocabulary(OOV) characters resulting from data cleaning, as discussed in Section 2. The presence of large out-of-vocabulary(OOV) characters in the noisy social media test set leads to the lowest accuracy of 26.87% achieved by the multilingual IndicXlit model.

6. **Enhancement through Fine-Tuning:** Finally, we observed that the fine-tuning process did not lead to any significant enhancement in transliteration performance. Specifically, when fine-tuning the model trained with the clean dataset on our noisy set, there was a decrease in accuracy from 41.57% in setup 1 to 36.63% in setup 12. On the other hand, fine-tuning the multilingual IndicXlit model with our noisy dataset resulted in a notable improvement in accuracy, increasing from 26.87% in setup 5 to 38.05% in setup 13, yet still lower than the 41.57% accuracy achieved in setup 1. Similarly, when fine-tuning the Noisy model of setup 1 on the clean dataset in setup 14, a comparable accuracy of 77.07% was attained similar to the 77.78% accuracy observed in setup 7. However, when the same model evaluated with the noisy test set in setup 15, the accuracy dropped down to 28.16%.

In summary, we have not found a single transliteration setup that performs well on both the noisy and clean test sets. However, among all the fifteen setups, the merge model setup with unique pairs (MergeUniqueTrans, highlighted with bold) achieved reasonably good performance in handling both types of data together as shown in Table 3. Additionally, fine-tuning the multilingual IndicXlit model on the noisy dataset yielded better performance than evaluating the IndicXlit model directly on the noisy test set. This indicates that training a multilingual transliteration model by combining both noisy and clean datasets can potentially enhance performance and effectively handle both the noisy and clean datasets simultaneously.

5 Error Analysis

This section presents key observations derived from the transliteration output, as shown in Table 4. Combining the noisy and clean datasets in setup 3 and training the transliteration model with unique pairs resulted in improved performance compared to the solely noisy dataset-trained model in setup 1 while tested on the noisy test set. This finding is interesting as it highlights our efforts to develop models capable of handling both noisy and clean datasets. Remarkably, the same merged model in setup 9, containing unique clean and noisy pairs, achieved comparable results to setup 7 when tested with the clean test set. To gain deeper insights, we conducted an error analysis on transliteration outputs comparing the Noisy model in setup 1 to the Merge model in setup 3. Additionally, the state-of-the-art multilingual IndicXlit model exhibited limited performance on the noisy test set in setup 5 compared to the noisy dataset-trained model in setup 1. We conducted a similar error analysis for the setup 5 transliteration output to explore this further. We marked the transliteration errors with red in Table 4.

Table 4: In-depth error analysis on transliteration output from setup 1, setup 3, and setup 5

Input (Roman)	**Native (Ground truth)**	**Merged Model Output (*setup 3*)**	**Noisy Model Output (*setup 1*)**	**IndicXlit Model Output (*setup 5*)**
dhekeri	ঢেকেৰী	ঢেকেৰী	ধেকেৰী	ঢেকেৰী
uddipona	উদ্দীপনা	উদ্দীপনা	উদিপনা	উদ্দীপনা
montra	মন্ত্র	মন্ত্র	মনত্র	মন্ত্রা
ahom	আহোম	আহোম	অসম	আহম
sandubi	চানডুবি	চানডুবি	চান্দুবি	চান্দুবি
ingland	ইংলেণ্ড	ইংলেণ্ড	ইংলান্দ	ইংলেণ্ড
2mak	তোমাক	তোমাক	তোমাক	no output
swali**2**r	ছোৱালীটোৰ	ছোৱালীটোৰ	ছোৱালীটোৰ	no output
axom'ot	অসমত	অসমত	অসমত	no output

Our observations revealed that the transliteration model trained solely on noisy social media text in setup 1 encountered challenges in handling *consonant conjuncts* and *named entities*, as evident in Table 4. Conversely, the merge model, trained with a combination of both noisy and clean datasets in setup 3, demonstrated superior performance in transliterating *consonant conjuncts* and *named entities*. Additionally, both the noisy model of setup 1 and the merged model of setup 3 correctly transliterated *letter-number mixing* and the use of *apostrophes* (') inside words, which are common practices in social media texts. However, the multilingual IndicXlit model, primarily trained on clean/canonical datasets, struggled to cope with such noisy behavior and treated these as out-of-vocabulary (OOV) characters, resulting in no transliteration outputs.

6 Related Work

The transliteration of informal text, particularly informal Romanized Indian language text commonly found in social media, has become a prominent research topic, especially in the context of shared tasks organized by the Forum for Information Retrieval (FIRE). Notably, the FIRE 2013 and FIRE 2014 shared tasks used song lyrics data written in Roman script for evaluation and analysis. Participants were challenged to develop independent transliteration models for specific subtasks using provided fixed datasets(Roy et al. (2013), Joshi et al. (2013), Pakray and Bhaskar (2013), Bhat et al. (2014), Mukherjee et al. (2014), Ganguly et al. (2014), Sapkal and Shrawankar (2016)).

In recent years, significant progress has been made in the transliteration domain. For instance, for processing canonical Indic language transliteration pairs, Roark et al. (2020) released the Dakshina dataset, comprising 12 South Asian languages (mainly Indian languages) written in Latin script, and provided baseline results for transliteration and language modeling tasks. Kunchukuttan et al. (2021) evaluated neural machine transliteration for English and 10 Indian languages, focusing on multilingual transliteration to leverage orthographic similarity between Indian languages. Following this work, Madhani et al. (2022) introduced the Aksharantar dataset, a large transliteration dataset for 21 Indian languages, and achieved state-of-the-art results using a single transformer-based multilingual transliteration model, IndicXlit. However, it is worth noting that none of the recent work has specifically addressed the transliteration problem for Romanized social media datasets.

7 Conclusion and Future Work

In conclusion, our study assessed the performance of the state-of-the-art machine transliteration model in handling both non-canonical social media text and manually forward transliterated canonical data within the context of the Assamese language. Our investigations across 15 different setups revealed that models trained solely on canonical data encountered challenges with ambiguity and noise present in social media text, while models trained solely on noisy data faced limitations with clean canonical data. However, the combined model trained on both canonical and non-canonical data demonstrated comparable results, emphasizing the need for robust models that can effectively handle both clean and noisy transliterations.

In the future, our main focus will be on enhancing the transliteration model to effectively handle both canonical and non-canonical data. Our primary challenge will lie in dealing with the complexities associated with processing real-world noisy social media data in non-canonical form. Additionally, exploring the transliteration of not only non-canonical romanized Assamese data but also investigating the processing of code-mixed Assamese data represents a potential future direction. Given the linguistic diversity in India, addressing this transliteration challenge holds immense importance across various Indian social media scenarios and languages. Furthermore, extending this study to train a multilingual transliteration model that incorporates both canonical and non-canonical data in a unified system shows promising potential for advancing natural language processing across a wide range of linguistic contexts.

8 Acknowledgements

We are thankful to Open Source Intelligence Lab, Indian Institute of Technology, Guwahati for providing the dataset. This work is partially funded by the Ministry of Electronics & Information Technology, Government of India.

References

Rafael E. Banchs, Min Zhang, Xiangyu Duan, Haizhou Li, and A. Kumaran. 2015. Report of NEWS 2015 machine transliteration shared task. In *Proceedings of the Fifth Named Entity Workshop*, pages 10–23, Beijing, China. Association for Computational Linguistics.

Irshad Ahmad Bhat, Vandan Mujadia, Aniruddha Tammewar, Riyaz Ahmad Bhat, and Manish Shrivastava. 2014. Iiit-h system submission for fire2014 shared task on transliterated search. In *Proceedings of the 6th Annual Meeting of the Forum for Information Retrieval Evaluation*, FIRE '14, page 4853, New York, NY, USA. Association for Computing Machinery.

C Chandramouli and Registrar General. 2011. Census of india. *Rural Urban Distribution of Population, Provisional Population Total. New Delhi: Office of the Registrar General and Census Commissioner, India*.

Debasis Ganguly, Santanu Pal, and Gareth J. F. Jones. 2014. Dcu@fire-2014: Fuzzy queries with rule-based normalization for mixed script information retrieval. In *Proceedings of the 6th Annual Meeting of the Forum for Information Retrieval Evaluation*,

FIRE '14, page 8085, New York, NY, USA. Association for Computing Machinery.

Hardik Joshi, Apurva Bhatt, and Honey Patel. 2013. Transliterated search using syllabification approach. In *Forum for information retrieval evaluation*.

Kevin Knight and Jonathan Graehl. 1997. Machine transliteration. *arXiv preprint cmp-lg/9704003*.

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.

Yash Madhani, Sushane Parthan, Priyanka Bedekar, Ruchi Khapra, Vivek Seshadri, Anoop Kunchukuttan, Pratyush Kumar, and Mitesh M. Khapra. 2022. Aksharantar: Towards building open transliteration tools for the next billion users.

Abhinav Mukherjee, Anirudh Ravi, and Kaustav Datta. 2014. Mixed-script query labelling using supervised learning and ad hoc retrieval using sub word indexing. In *Proceedings of the 6th Annual Meeting of the Forum for Information Retrieval Evaluation*, FIRE '14, page 8690, New York, NY, USA. Association for Computing Machinery.

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 (Demonstrations)*, pages 48–53, Minneapolis, Minnesota. Association for Computational Linguistics.

Partha Pakray and Pinaki Bhaskar. 2013. Transliterated search system for indian languages. In *Preproceedings of the 5th FIRE-2013 workshop, forum for information retrieval evaluation (FIRE)*.

Brian Roark, Lawrence Wolf-Sonkin, Christo Kirov, Sabrina J. Mielke, Cibu Johny, Isin Demirsahin, and Keith Hall. 2020. Processing South Asian languages written in the Latin script: the Dakshina dataset. In *Proceedings of the Twelfth Language Resources and Evaluation Conference*, pages 2413–2423, Marseille, France. 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 *Proceedings of the 4th and 5th Annual Meetings of the Forum for Information Retrieval Evaluation*, FIRE '12 & '13, New York, NY, USA. Association for Computing Machinery.

Krutika Sapkal and Urmila Shrawankar. 2016. Transliteration of secured sms to indian regional language. *Procedia Computer Science*, 78:748–755.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

G Vyshnavi, G Sridevi, P Krishna Reddy, and M Ritesh. 2022. Syllabified sequence-to-sequence attention for transliteration. In *Proceedings of the 36th Pacific Asia Conference on Language, Information and Computation*, pages 617–625.