A Benchmark and Dataset for Post-OCR text correction in Sanskrit

Ayush Maheshwari[1], **Nikhil Singh**[*], **Amrith Krishna**[2] and **Ganesh Ramakrishnan**[1]
{ayusham,ganesh}@cse.iitb.ac.in,
{nikhil3198,krishnamrith12}@gmail.com
[1]Indian Institute of Technology Bombay, [2]Uniphore

Abstract

Sanskrit is a classical language with about 30 million extant manuscripts fit for digitisation, available in written, printed or scanned-image forms. However, it is still considered to be a low-resource language when it comes to available digital resources. In this work, we release a post-OCR text correction dataset containing around 218,000 sentences, with 1.5 million words, from 30 different books. Texts in Sanskrit are known to be diverse in terms of their linguistic and stylistic usage since Sanskrit was the 'lingua francua' for discourse in the Indian subcontinent for about 3 millennia. Keeping this in mind, we release a multi-domain dataset, from areas as diverse as astronomy, medicine and mathematics, with some of them as old as 18 centuries. Further, we release multiple strong baselines as benchmarks for the task, based on pre-trained Seq2Seq language models. We find that our best-performing model, consisting of byte level tokenization in conjunction with phonetic encoding (Byt5+SLP1), yields a 23% point increase over the OCR output in terms of word and character error rates. Moreover, we perform extensive experiments in evaluating these models on their performance and analyse common causes of mispredictions both at the graphemic and lexical levels. Our code and dataset is publicly available at https://github.com/ayushbits/pe-ocr-sanskrit.

1 Introduction

Post-OCR text correction is a crucial post-processing step employed for correcting errors from the predictions of Optical Character Recognition (OCR) systems (Rijhwani et al., 2021a). A post-OCR corrector leverages the distributional information encoded in language models that aims to not only handle the systemic errors introduced by the OCR engine but also to predict meaningful and fluent sequences based on the context (Saluja et al., 2019). For a language like Sanskrit, the sources of OCR errors are diverse, owing to the availability of printed historical documents that vary vastly on a number of factors such as scan quality, book layout, typefaces, the orthographic similarity of letters in the alphabet, *etc.* (see Figure 1). Moreover, processing texts in Sanskrit is often challenging as the language is morphologically rich, lexically productive, follows relatively free-word order and is a low-resource language with limited available machine-readable corpora (Krishna et al., 2021).

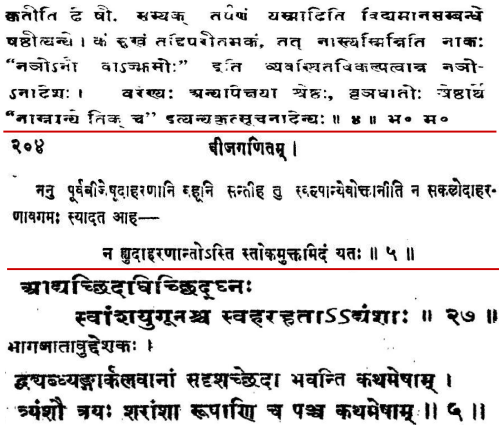

Figure 1: Image samples from different pages of our dataset.

In this work, we release a large Sanskrit post-correction dataset of more than 218,000 manually verified sentences, consisting of 1.5 million words. Our dataset consists of sentences from 30 books from domains as diverse as philosophy, literature, astronomy, medicine, mathematics *etc.*. Figure 1 shows a sample of the scanned images from these books, from which we obtained our dataset. Further, the sample clearly demonstrates the diversity in some of the aforementioned factors affecting the quality of the OCR predictions. Histori-

[*]Work done while interning at IIT Bombay.

cally, depending on the region or the time period in which it was used, several writing systems and scripts were employed for writing Sanskrit. However, the advent of the printing press largely standardized the use of the 'Devanāgari' script as the default writing system for Sanskrit.

We additionally release a set of strong seq2seq baselines to benchmark for the task, including a CopyNet based LSTM model (Gu et al., 2016) and four pretrained seq2seq systems (LMs). We find that all the pretrained-LM based baselines improve over the predictions from the original OCR. The best model which invokes byte level tokenization, *viz.*, ByT5 (Xue et al., 2022), in conjunction with phonetic encoding (SLP1), among these benchmarks (all described in § 3) reports a character and word error rates of 2.98% and 23.19% respectively, as against that of 3.89% and 30.23% from the original OCR. This is primarily due to the ability of byte-level tokenizers to learn arbitrarily longer text in a setting where the frequency of words is low and out-of-vocabulary words are high (§2). Moreover, this goes well with the fact that the writings in Sanskrit follow a phonemic orthography, i.e. phonemes have a direct one-to-one correspondence with the orthographic symbols. We identify that most errors arising from the original OCR are from mispredictions in word boundary detection, diacritics and orthographically similar characters. Further, as the performance of the post-OCR text correction system is highly dependent on the predictions of the OCR, we also release the test dataset used for testing our current OCR. The test dataset consists of 500 images and their corresponding text, which can be used to benchmark an OCR, prior to using its predictions for post-OCR text correction.

2 Dataset

Sanskrit used to be the 'lingua franca' for scholarly discourse in the Indian subcontinent for about three millennia and the classical language is still in sustenance in the region. It is estimated that as many as 30 million extant documents, more than that in Greek and Latin combined, are fit for digitisation in Sanskrit (Goyal et al., 2012; Adiga et al., 2021). The current corpus is released as part of our attempt at large scale digitisation of old manuscripts in Sanskrit. Our corpus contains about 30 books, a subset of 103 books in our digitisation pipeline. These books were originally published at least a century ago, and are manually verified to have no copyright issues. We consider printed versions of these books, most of them reprinted in the first half of the twentieth century. While, these books are widely accessible to the public via libraries and academic institutions, we manually had to scan several of them as part of its digitisation process. These books vary widely in their vocabulary and stylistic usage owing to the differences in the domain and the original time period of publication, where the latter can be as old as the fifth century AD.

Devanāgari	क्	क	का	कि	की	कृ	क्क	अ
Romanized	k	ka	kā	ki	kī	kr̥	kka	a

Figure 2: Devanāgari and Romanised representation of 'k' followed by different vowels. Conjunct consonant ('kka') may also have separate symbols in Devanāgari.

Split	# sentences	# words
Train	208,173	1,444,913
Validation	5000	34,762
Test	5000	34,705

Table 1: Number of words and sentences in the dataset split.

We release a multi-domain dataset from 30 different books and have 218,000 manually verified sentences in it. The share of each book in the corpus amounts to 3.33% on average with a variance of 4.09, in terms of the number of pages. The corpus consists of more than 1.5 million tokens, with an average frequency of 2.59, and has a vocabulary of 581,445 unique words. Further, 88% has a frequency of one and more than 96% of the words appear less than 5 times. Such a frequency distribution of tokens in Sanskrit corpora is common, given the morphological richness and lexical productivity (due to compounding) in Sanskrit (Krishna et al., 2017; Hellwig, 2010-2016). Further, the average word length is 10.4 characters. In Table 1, we present count of sentences and tokens in our train-test-val split. Of the 20,738 words in the test data vocabulary, 54.53% of those are out of vocabulary. Earlier approaches to post-OCR text correction have employed lexicon-driven approaches for several languages, though such approaches without a wide coverage lexicon might be challenging for a language like Sanskrit (Bassil and Alwani, 2012; Carlson and Fette, 2007).

Prior work in post-OCR text correction in Sanskrit (Krishna et al., 2018) focused on texts written in IAST or Romanized Sanskrit (Monier-Williams et al., 1899), while the current dataset is focused on Devanāgari. All the books we consider here use Devanāgari script as the writing system. While Devanāgari is in existence since the fourth century CE (State), 1896), it has become the primary standard for writing Sanskrit, and several other languages, with the advent of the printing press in India. The script consists of 47 primary characters and is a left-to-right abiguda, where contiguous consonant-vowel sequences are treated as unitary units. As shown in Figure 2, the vowels following the consonants (k in the figure), are treated as secondary units. These units are expressed as 'mātras' in the writing system, which essentially are diacritic markers. These markers may appear before, after, above or below an orthographic consonant symbol as shown in Figure 2. The same vowel, say '*a*' in the romanised script, is written as a different character when used independently as a primary unit. Similarly, conjunct consonants, like '*kka*' in the figure, would also result in a different orthographic unit, leading to an increase in possible output units for the original OCR. Moreover, these orthographically similar units can also be confusing to an OCR system.

2.1 OCR Editing Process

The current work is part of an OCR project that aims to digitize hundreds of Sanskrit books present in scanned image format[1]. The dataset is an outcome of a publicly funded project, primarily carried out by researchers at IIT Bombay. The project currently has 103 books in its pipeline. Our dataset consists of books primarily from philosophy, literature, mathematics. medicine and astronomy. List of books is provided in Figure 4 in appendix.

To aid the process of correction of OCR output, we developed an open-source post-OCR editing tool (Maheshwari et al., 2022) that reduces the cognitive and editing load of the users and increases the speed of text correction. The in-house developed tool is used for OCR correction, verification and proofreading. Currently, 14 experts contribute to various stages of the digitisation process. These experts are either linguists, trained specifically in Sanskrit linguistics, or computational linguists, and seven of them are working full time for the project. Each page in the book passes through a three step process, and a separate expert oversees each step. The 3 steps are: 1) Manual correction/post-editing of OCR prediction by looking at the original scanned image, 2) Verification of the corrected text performed in the previous step, and 3) Proofreading of the text to check for obvious errors. Verification is primarily aimed at maintaining fidelity of the corrected text to the scanned lines and proofreading is aimed at ensuring linguistic and semantic correctness of the text.

3 System Descriptions

OCR post-correction is a text correction task which can be formalised as a monotone seq2seq model (Schnober et al., 2016). We use an encoder-decoder framework that takes predictions from an OCR as its input. While we use multiple pretrained seq2seq models as our baselines, none of these has Sanskrit as one of their languages. However, Devanāgari script is employed in other languages, such as Hindi, which are present in these models. Secondly, unicode encoding of Devanāgari often poses several challenges owing to the variable byte length employed per character for encoding. Hence, we losslessly transliterate the text into SLP1, an ASCII-based case-sensitive transliteration scheme in our experiments.

Baseline OCR Model : Our baseline OCR model is an OCR engine that uses the Tesseract OCR (Smith, 2007). The model is fine-tuned upon 20,000 synthetically-created images with the Sanskrit language flag. We release our OCR test set along with the post-OCR correction dataset.

CopyNet (Gu et al., 2016): uses a copying mechanism in an LSTM-based seq2seq framework to leverage the (partial) overlap between input and output strings. The model consists of 3 LSTM modules stacked on top of each other for both the encoder and decoder. Following Krishna et al. (2018), we use BPE for learning the vocabulary, which has shown the ability to handle corpora with 'rare words' (Sennrich et al., 2016).

mBART (Liu et al., 2020) is a multilingual variant of the BART, both of which are seq2seq models. It has an autoregressive decoder and a BERT-based encoder. We used mBART-50 (Tang et al., 2020), specifically its *HuggingFace* implementa-

[1]Project website: https://www.cse.iitb.ac.in/~ocr/

tion (large), in our experiments which has been trained on large monolingual corpus of 50 languages. Here, we use text in its original form as well as in the transliterated SLP1 form.

mT5 (Xue et al., 2021) is a multilingual variant of T5 (Raffel et al., 2020), trained on 107 languages. T5 is a seq2seq text generation model, pretrained on a mixture of supervised and unsupervised tasks using a span-corruption objective. In experiments, we employ the mT5 base model from *HuggingFace* along with BPE tokenization and use both devānagari and SLP1 encoded text.

ByT5 (Xue et al., 2022): Given that we have a corpus with mostly 'rare words' (*c.f.*, section 2), any unseen set in Sanskrit will suffer from out-of-vocabulary words. A natural solution is to tokenize words at a character level where each character is represented by UTF-8 bytes. ByT5 is a variant of mT5 except that model is fed with a fixed 256 byte values. In experiments, we use ByT5 small model from *HuggingFace* with byte tokenizer and SLP1 encoded text.

IndicBART (Dabre et al., 2021): It is a multilingual BART-based model trained on 11 Indic-family languages in devānagari script. The model is roughly half the size of mT5 model.

4 Experiments and Results

In Table 2, we present the macro-averaged Word Error Rate (WER) and Character Error Rate (CER) for each of our baseline systems. The predictions directly from OCR report a CER and WER of 3.89% and 30.23% respectively. In our experiments, CopyNet's predictions worsen as per both our metrics, resulting in a CER and WER of 13.25% and 50.38% respectively. However, all of the pre-trained language model configurations employed for post-OCR correction improves over the original OCR predictions. In general, we find that use of Devanāgari scripts instead of SLP1 to encode text in Sanskrit, results in improved performance for the task. However, with ByT5, we find that our model produces truncated outputs mostly due to an increase in sequence length in ByT5 due to the byte-level vocabulary used in it. The output from ByT5-Dev has a CER of 6.17% and a WER of 27.72%, higher than that of ByT5-SLP1, when a sequence length of 1024 was used. Even though the CER is further reduced to 4.59 (from 6.17) for ByT5-Dev, when its maximum sequence length is reconfigured to 2048, it still does not outperform ByT5-SLP1 (refer Table 3). The use of SLP1 encoding for Sanskrit converts them to ASCII sequences, thereby reducing the overall sequence length for input. The ByT5 configuration with SLP1 encoding of text currently yields the best outcome in our experiments. We discuss the impact of different sequence length and memory overheads between Dev and SLP1 variants of ByT5 in Appendix A.

Encoding	Model	CER	WER
Dev	OCR	3.89	30.23
Dev	mBART	3.50 (+10)	26.11 (+13.7)
SLP1	mBART	3.71 (+4.5)	26.60 (+12)
Dev	IndicBART	3.55 (+8.7)	25.73 (+14.9)
Dev	CopyNet	13.25 (-240)	50.38 (-66)
SLP1	mT5	3.53 (+9.2)	26.47 (+12.5)
Dev	mT5	3.34 (+14.1)	25.57 (+15.4)
SLP1	ByT5	**2.98** (+23.4)	**23.19** (+23.3)

Table 2: CER and WER (lower is better) on Post-OCR correction task for different encoding schemes and model. Numbers in brackets () corresponds to percentage improvement over OCR model output (top row). All methods are evaluated on devānagari text and all models except ByT5 uses BPE tokenizer. Dev refers to devānagari.

We observe three primary sources of errors from the original OCR predictions, namely, word and sentence boundary identification owing to missing or extraneous prediction of space and sentence boundary markers, mispredictions due to mātra or diacritics, and mispredictions arising out of orthographically similar characters. All of these cumulatively contribute to 61.76% of the character-level errors. Boundary detection at the word level and at the sentence level, identified by a space marker or by sentence terminating punctuation, contributes to 26.96% of the OCR errors. 89.3% of the boundary detection errors arise out of identifying word boundaries. Similarly, errors due to incorrect or missing mātras (diacritics) contribute to 22.41% of all the errors in OCR. In Sanskrit, these mātras are generally secondary vowels following a consonant, a phenomenon common in abiguda writing systems (§2). Mispredictions specifically due to orthographically similar characters contribute to 12.39% of the total errors. With ByT5, the best performing model we report, we find an error reduction of 44.45%, 37.74% and 14.16% reduction in boundary identification, mātra prediction

and error corrections from orthographically similar predictions respectively.

As a further analysis, we collect the most frequent 300 tokens in the corpus, with at least three letters for a word, and find a total of 2875 occurrences in the ground truth corpus. Among these frequent tokens, OCR predicts each of them correctly at least once. Further, to identify possible similar tokens predicted instead of the correct token, we use the Ratcliff pattern recognition algorithm (Black, 2004), with a matching ratio of 0.6. Here, we find that 8.27% of the token occurrences among the most frequent tokens do not have a corresponding prediction that satisfies the matching criteria. With ByT5, this number is reduced by 2.09%. Moreover, in both OCR predictions and ByT5 based predictions, we mostly find unique patterns in mispredictions for each token and are not able to find any consistent or systemic patterns for each token.

4.1 Experiments on out of domain test dataset

Similar to prior works (Rijhwani et al., 2021b; Krishna et al., 2017), we ensure that there is no sentence-level (sequence) overlap between the train, test and validation split. Though there is an overlap in terms of the books, we ensure that none of the test-data sentences are seen during training. To test the generalizibility of our models, we use an out-of-domain test data comes from a completely new book, Brihat-samhita, not included in any of the train-test-validation splits. We also release a new out-of-domain test dataset which comes from a text that is not part of any of the 30 texts included in our dataset. Figure 3 shows the performance of all the OCR, ByT5 and the mT5 systems for this dataset. Here, ByT5 has shown to significantly reduce the CER and WER from the OCR outputs.

5 Conclusion

We release a dataset consisting of 218,000 sentences from 30 books for Sanskrit Post-OCR text correction. We also release a set of strong baselines as a benchmark, which currently shows consistent and significant improvements from the OCR predictions, both on the in-domain test data and out-of-domain test data. All our baselines, in spite of not seeing Sanskrit during pretraining, have shown to generalise well for the task. While

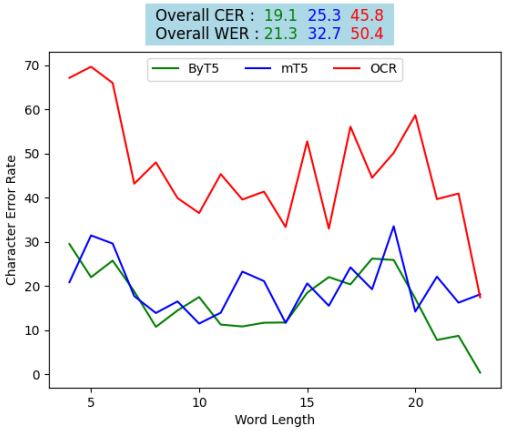

Figure 3: Comparison of CER with different word lengths on an out-of-corpus test set of 500 sentences.

using Devanāgari Unicode encoding for our experiments has shown to perform better than using SLP1 for multiple baselines, SLP1-based encoding on ByT5 gives the best performance overall.

6 Limitations

A major limitation with the current baselines is the mispredictions happening at the word level. Here, of the mispredicted words by ByT5-SLP, our best performing model, 71.17% are not even valid words in Sanskrit. None of our pretrained models currently are lexically or morphologically aware resulting in the formation of invalid words in the language. Moreover, owing to the low-resource nature of the language, none of the pretrained language models we employed used Sanskrit for its pretraining. An immediate challenge with outputs of our post-OCR text correction would be the use of these predictions for downstream tasks, which are heavily reliant on rule-based morphological analysis of these words (Krishna et al., 2021). We plan to incorporate morphologically aware self-training approaches and dynamic markup decoding (De Cao et al., 2021) which can incorporate various valid inflected forms of a stem in a trie to handle such scenarios.

7 Acknowledgements

We thank anonymous reviewers for providing constructive feedback. Ayush Maheshwari is supported by a Fellowship from Ekal Foundation (www.ekal.org). Ganesh Ramakrishnan is grateful to NLTM OCR Bhashini project as well as the IIT Bombay Institute Chair Professorship for their support and sponsorship.

References

Devaraja Adiga, Rishabh Kumar, Amrith Krishna, Preethi Jyothi, Ganesh Ramakrishnan, and Pawan Goyal. 2021. Automatic speech recognition in sanskrit: A new speech corpus and modelling insights. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 5039–5050.

Youssef Bassil and Mohammad Alwani. 2012. Ocr context-sensitive error correction based on google web 1t 5-gram data set. *arXiv preprint arXiv:1204.0188*.

Paul E Black. 2004. Ratcliff/obershelp pattern recognition. *Dictionary of algorithms and data structures*, 17.

Andrew Carlson and Ian Fette. 2007. Memory-based context-sensitive spelling correction at web scale. In *Sixth International Conference on Machine Learning and Applications (ICMLA 2007)*, pages 166–171. IEEE.

Raj Dabre, Himani Shrotriya, Anoop Kunchukuttan, Ratish Puduppully, Mitesh M Khapra, and Pratyush Kumar. 2021. Indicbart: A pre-trained model for natural language generation of indic languages. *arXiv preprint arXiv:2109.02903*.

Nicola De Cao, Gautier Izacard, Sebastian Riedel, and Fabio Petroni. 2021. Autoregressive entity retrieval. In *International Conference on Learning Representations*.

Pawan Goyal, Gérard Huet, Amba Kulkarni, Peter Scharf, and Ralph Bunker. 2012. A distributed platform for Sanskrit processing. In *Proceedings of COLING 2012*, pages 1011–1028, Mumbai, India. The COLING 2012 Organizing Committee.

Jiatao Gu, Zhengdong Lu, Hang Li, and Victor OK Li. 2016. Incorporating copying mechanism in sequence-to-sequence learning. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1631–1640.

Oliver Hellwig. 2010-2016. *DCS - The Digital Corpus of Sanskrit*. Berlin.

Amrith Krishna, Bodhisattwa P Majumder, Rajesh Bhat, and Pawan Goyal. 2018. Upcycle your ocr: Reusing ocrs for post-ocr text correction in romanised sanskrit. In *Proceedings of the 22nd Conference on Computational Natural Language Learning*, pages 345–355.

Amrith Krishna, Bishal Santra, Ashim Gupta, Pavankumar Satuluri, and Pawan Goyal. 2021. A Graph-Based Framework for Structured Prediction Tasks in Sanskrit. *Computational Linguistics*, 46(4):785–845.

Amrith Krishna, Pavan Kumar Satuluri, and Pawan Goyal. 2017. A dataset for Sanskrit word segmentation. In *Proceedings of the Joint SIGHUM Workshop on Computational Linguistics for Cultural Heritage, Social Sciences, Humanities and Literature*, pages 105–114, Vancouver, Canada. Association for Computational Linguistics.

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

Ayush Maheshwari, Ajay Ravindran, Venkatapathy Subramanian, Akshay Jalan, and Ganesh Ramakrishnan. 2022. Udaan-machine learning based post-editing tool for document translation. *arXiv preprint arXiv:2203.01644*.

Monier Monier-Williams, Ernst Leumann, and Carl Cappeller. 1899. A sanskrit-english dictionary: etymologically and philologically arranged with special reference to cognate indo-european languages.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, Peter J Liu, et al. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *J. Mach. Learn. Res.*, 21(140):1–67.

Shruti Rijhwani, Daisy Rosenblum, Antonios Anastasopoulos, and Graham Neubig. 2021a. Lexically aware semi-supervised learning for ocr post-correction. *Transactions of the Association for Computational Linguistics*, 9:1285–1302.

Shruti Rijhwani, Daisy Rosenblum, Antonios Anastasopoulos, and Graham Neubig. 2021b. Lexically aware semi-supervised learning for OCR post-correction. *Transactions of the Association for Computational Linguistics*, 9:1285–1302.

Rohit Saluja, Ayush Maheshwari, Ganesh Ramakrishnan, Parag Chaudhuri, and Mark Carman. 2019. Ocr on-the-go: Robust end-to-end systems for reading license plates & street signs. In *2019 International Conference on Document Analysis and Recognition (ICDAR)*, pages 154–159. IEEE.

Carsten Schnober, Steffen Eger, Erik-Lân Do Dinh, and Iryna Gurevych. 2016. Still not there? comparing traditional sequence-to-sequence models to encoder-decoder neural networks on monotone string translation tasks. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 1703–1714, Osaka, Japan. The COLING 2016 Organizing Committee.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational*

Linguistics (Volume 1: Long Papers), pages 1715–1725, Berlin, Germany. Association for Computational Linguistics.

Ray Smith. 2007. An overview of the tesseract ocr engine. In *Ninth international conference on document analysis and recognition (ICDAR 2007)*, volume 2, pages 629–633. IEEE.

Bombay (India : State). 1896. *Gazetteer of the Bombay Presidency*. v. 1, pt. 1. Printed at the Government Central Press.

Yuqing Tang, Chau Tran, Xian Li, Peng-Jen Chen, Naman Goyal, Vishrav Chaudhary, Jiatao Gu, and Angela Fan. 2020. Multilingual translation with extensible multilingual pretraining and finetuning. *arXiv preprint arXiv:2008.00401*.

Linting Xue, Aditya Barua, Noah Constant, Rami Al-Rfou, Sharan Narang, Mihir Kale, Adam Roberts, and Colin Raffel. 2022. Byt5: Towards a token-free future with pre-trained byte-to-byte models. *Transactions of the Association for Computational Linguistics*, 10:291–306.

Linting Xue, Noah Constant, Adam Roberts, Mihir Kale, Rami Al-Rfou, Aditya Siddhant, Aditya Barua, and Colin Raffel. 2021. mt5: A massively multilingual pre-trained text-to-text transformer. In *NAACL-HLT*.

Max token	2048		1024	
Model	**CER**	**WER**	**CER**	**WER**
OCR	4.02	30.7	4.67	30.02
ByT5-Dev	4.59	26.6	13.12	38.6
ByT5-SLP1	**3.05**	**23.5**	**3.49**	**25.3**

Table 3: ByT5-Dev and SLP1 models are trained with different maximum token length. Max token size of 2048 is equivalent to 135 characters while max token size of 1024 is equivalent to 56 characters. We truncated the maximum character length in the test set corresponding to each experiment.

Appendix

A Impact of difference sequence length

ByT5 splits each character into bytes. Since the unicode encoding of Devanāgari characters typically have higher byte lengths, an input sequence in ByT5 often tends to be shorter, affecting contextual information in longer sentences. We present corresponding results in Table 3.

B List of Books

We present list of books used in our dataset in Table 4.

Book Name	Genre
Uttararamacharita by Bhavabhuti - commentary by Veeraraghava	Arts
Grahalaghava of Ganesh Daivajna	Astronomy
Suryasiddhanta of Ranganatha	Astronomy
Mahabhaskariyam	Astronomy
Aryabhatiya Bhashya of Gargyakerala Nilakantha Samabasiva Sastri K. Vol 1	Astronomy
Aryabhatiya Bhashya of Gargyakerala Nilakantha Samabasiva Sastri K. Vol 2	Astronomy
Aryabhatiya Bhashya of Gargyakerala Nilakantha Samabasiva Sastri K. Vol 3	Astronomy
Karana-Kutuhalam	Astronomy
LaghuManasa	Astronomy
Aryabhatiya commentary by Suryadeva Yajvan	Astronomy
Khandakhadyaka	Astronomy
Ganak Tarangini	Astronomy
Bijaganita with Navankjura-Apte	Mathematics
Bijaganitavatamksa of Narayan Shukla	Mathematics
Bijaganita with Bijankura	Mathematics
Ganitakaumudi of Narayanapandita (vol. 1)	Mathematics
Rekhaganita of Jagannatha Vol. 2	Mathematics
Bijaganita by Tr Abhyankar	Mathematics
Laghubhaskariyam Part 2	Mathematics
Rekhaganita of Jagannatha Vol. 1	Mathematics
Lilavati with kriyakramakri	Mathematics
Patiganita of Sridhara	Mathematics
Brahmasphutasiddhanta of Brahmagupta	Mathematics
Laghubhaskariyam	Mathematics
Hathayogapradipika by Svatmarama	Medicine
Mimamsanyayaprakasha by Aapdeva	Philosophy
Shastradipika by Parthasarthi	Philosophy
Shabdashaktiprakashika by Jagdishtarkalankara	Philosophy
Prakaranapanchika-Shalikanatha	Philosopy

Table 4: List of books used in the experiments.