IIIT-INDIC-HW-WORDS: A Dataset for Indic Handwritten Text Recognition

by

Santhoshini Gongidi, C V Jawahar

in

International Conference on Document Analysis and Recognition
: 1
-15

Report No: IIIT/TR/2021/-1

Centre for Visual Information Technology
International Institute of Information Technology
Hyderabad - 500 032, INDIA
July 2021

IIIT-INDIC-HW-WORDS: A Dataset for Indic Handwritten Text Recognition

Santhoshini Gongidi[0000−0001−5566−1904] and C V Jawahar[0000−0001−6767−7057]

Centre for Visual Information Technology,
IIIT Hyderabad, India
santhoshini.gongidi@research.iiit.ac.in, jawahar@iiit.ac.in

Abstract. Handwritten text recognition (HTR) for Indian languages is not yet a well-studied problem. This is primarily due to the unavailability of large annotated datasets in the associated scripts. Existing datasets are small in size. They also use small lexicons. Such datasets are not sufficient to build robust solutions to HTR using modern machine learning techniques. In this work, we introduce a large-scale handwritten dataset for Indic scripts containing 868K handwritten instances written by 135 writers in 8 widely-used scripts. A comprehensive dataset of ten Indic scripts are derived by combining the newly introduced dataset with the earlier datasets developed for Devanagari (IIIT-HW-DEV) and Telugu (IIIT-HW-TELUGU), referred to as the IIIT-INDIC-HW-WORDS.

We further establish a high baseline for text recognition in eight Indic scripts. Our recognition scheme follows the contemporary design principles from other recognition literature, and yields competitive results on English. IIIT-INDIC-HW-WORDS along with the recognizers are available publicly[1]. We further (i) study the reasons for changes in HTR performance across scripts (ii) explore the utility of pre-training for Indic HTRs. We hope our efforts will catalyze research and fuel applications related to handwritten document understanding in Indic scripts.

Keywords: Indic Scripts · Handwritten Text Recognition · Pre-training

1 Introduction

In the last two decades, large digitization projects converted paper documents as well as ancient historical manuscripts into the digital forms. However, they remain often inaccessible due to the unavailability of robust handwritten text recognition (HTR) solutions. The capability to recognize handwritten text is fundamental to any modern document analysis systems. There is an immediate need to provide content-level access to the millions of manuscripts and personal journals, large court proceedings and also develop HTR applications to automate processing of medical transcripts, handwritten assessments etc. In recent years, efforts towards developing text recognition systems have advanced due to the success of deep neural networks [27,15] and the availability of annotated datasets.

[1] http://cvit.iiit.ac.in/research/projects/cvit-projects/iiit-indic-hw-words

This is especially true for Latin scripts [5,21,4]. The IAM [20] handwritten dataset introduced over two decades ago, the historic George Washington [10] dataset and the RIMES [14] dataset are some of the popularly known datasets for handwritten text recognition. These public datasets enabled research for Latin HTR. Even today, these datasets are still being utilized to study handwritten data.

Compared to Latin HTR, Indic HTR remains understudied due to a severe deficit of annotated resources in Indian languages. Unlike many other parts of the world, a wide variety of languages and scripts are used in India. Therefore, collecting sizeable handwritten datasets for multiple Indic scripts becomes challenging and expensive. Existing annotated datasets for Indic HTR are limited in size and scope. They are approximately 5× smaller than the Latin counterparts.

Recent progress in text recognition is mainly credited to the easy access to large annotated datasets. Through this work, we make an effort to bridge the gap between the state of the arts in Latin and Indian languages by introducing a handwritten dataset written in 8 Indian scripts. We introduce a dataset for Bengali, Gujarati, Gurumukhi, Kannada, Odia, Malayalam, Tamil, and Urdu scripts. This dataset along with the datasets IIIT-HW-DEV, IIIT-HW-TELUGU introduced by Dutta et al. [8,9] for Devanagari and Telugu scripts provide handwritten datasets for text recognition in all ten prominent scripts used in India. We refer to this collective dataset as the IIIT-INDIC-HW-WORDS. Fig. 1 gives a glimpse into the new dataset and the writing style in eight different Indic scripts. This new dataset contains 868K word instances written in 8 prominent Indic scripts by 135 writers. Possibly this is the first attempt in even attempting offline handwriting in some of these scripts.

We hope that our dataset provides the much-needed resources to the research community to develop Indic HTR and to build valuable applications around Indian languages. The diversity of the IIIT-INDIC-HW-WORDS dataset makes it possible to utilize this dataset for other document analysis problems also. Script identification, handwriting analysis, and synthesis for Indian languages are examples of other problems that can be enabled using this dataset. This dataset could also be beneficial for study of script-independent recognition architectures for HTR.

In this work, we also present a simple and effective text recognition architectures for Indian HTR. We establish a high baseline on scripts presented in the IIIT-INDIC-HW-WORDS. The results and related discussion are in Section 4. We also study the benefit of using architectures pre-trained on other scripts. Building robust recognizers in various scripts requires a large amount of data for each script. With transfer learning from other scripts, the excessive requirements of large dataset and training time can be reduced. We also investigate the relation between script similarity and pre-training. We explore both these ideas in Section 5.

The major contribution of this work is as follows:

i. We introduce a large dataset — IIIT-INDIC-HW-WORDS, consisting of annotated handwritten words written in 8 Indic scripts.

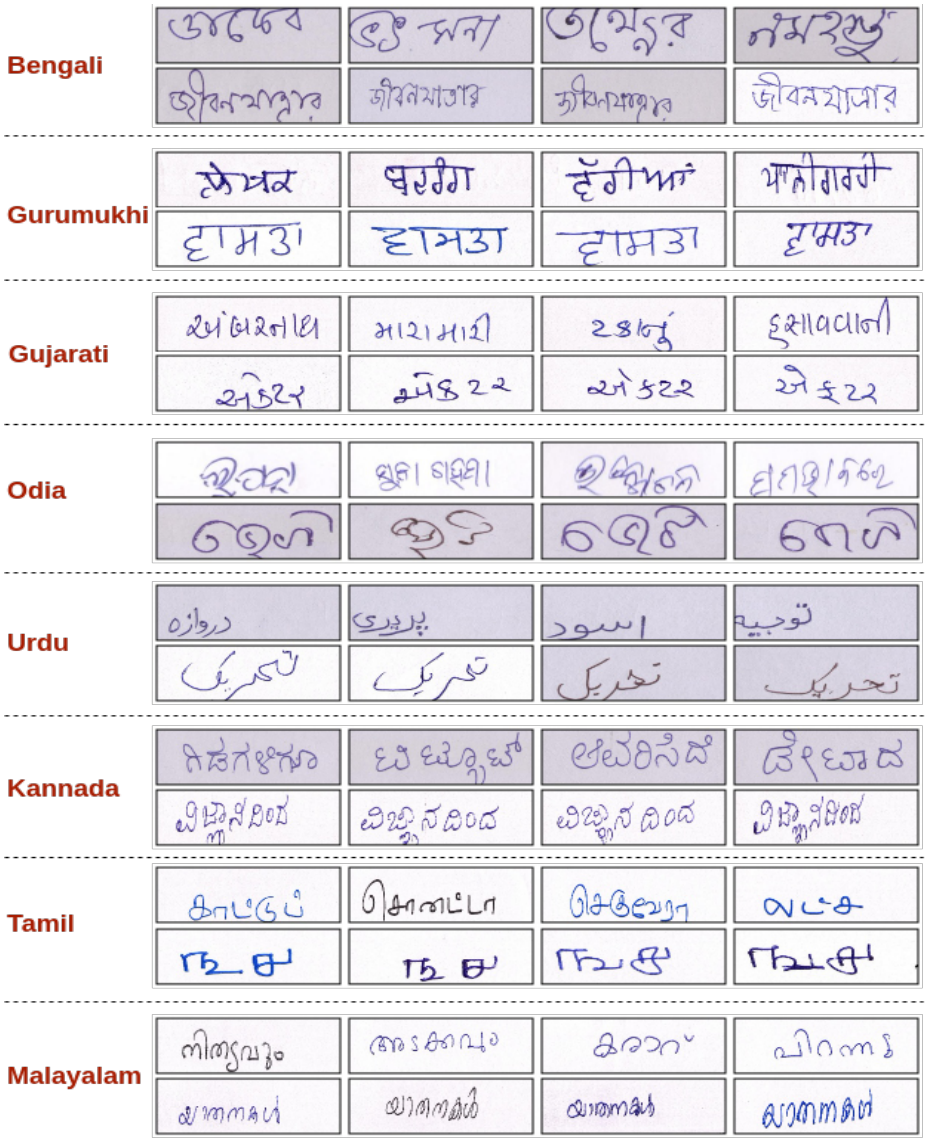


Fig. 1. Word instances from our IIIT-INDIC-HW-WORDS dataset. Out of the 10 major Indic scripts used, we present datasets for 8 scripts: Bengali, Gujarati, Gurumukhi, Kannada, Malayalam, Odia, Tamil, Urdu and complement the recent efforts in Dutta et al. [8] and [9] for Hindi and Telugu. For each script, row 1 shows writing style variations for a specific writer across different words. Row 2 images presented in a specific script block show four writing variations for a particular word.

ii. We establish a high baseline recognizer for Indic HTR. Our recognition algorithm is highly script independent. In this process, we identify the appropriate architecture for Indic scripts and establish benchmarks on the IIIT-INDIC-HW-WORDS dataset for future research.
iii. We explore the possibility of transferability across scripts with a set of systematic experiments.

2 Datasets for HTR

Approaches for recognizing natural offline handwriting are heavily data-driven today. Most of them use machine learning methods and learn from annotated examples. This demands for script/language-specific collection of examples. For machine learning methods to be effective, the annotated datasets should be substantial and voluminous in size and diverse in nature. For HTR, this implies that the dataset should have (i) many writers, (ii) extensive vocabulary, and (iii) huge number of samples.

Public datasets such as IAM [20] and George Washington [10] have catalyzed the research in handwriting recognition and retrieval in the past. Datasets for Latin scripts such as IAM [20] and Bentham collection [24] have over 100K running words with large lexicons. The IAM dataset is one of the most commonly used dataset in HTR for performance evaluation even today. This dataset contains 115,320 word instances written by 657 writers, and has a lexicon size of 10,841. This dataset provides annotations at word-level, line-level, and page-level. As the performance on IAM dataset has started to saturate, more challenging datasets have started to surface. The historical datasets such as Bentham collection [24] and the READ dataset [25] are associated with line-level and page-level transcriptions. Annotations at the line, word, and document level provide opportunities for developing methods that use language models and higher order cues. Indian languages are still in their infancy as far as HTRs are concerned, and we limit our attention to the creation of word-level annotations in this work.

Lack of large handwritten datasets remains as a major hurdle for the development of robust solutions to Indic HTR. Table 1 presents the list of publicly available datasets in Indian languages. We also contrast our newly introduced IIIT-INDIC-HW-WORDS dataset with the existing ones in size, vocabulary and the number of writers in Table 1. Most of the Indic handwritten datasets are smaller than the IAM in the number of word instances, writing styles, or lexicon size. For example, the lexicon used to build the CMATER2.1 [3] and the TAMIL-DB datasets [28] consist of city names only. The CENPARMI-U dataset provides only 57 different financial terms in the entire dataset. The small and restrictive nature of the lexicon for these datasets limits the utility while building a generic HTR to recognize text from a large corpus of words. ROYDB [22] and LAW datasets [19] use a larger lexicon. However, the size of these dataset is small, and possibly insufficient to capture the natural variability in handwriting. The PBOK [1] dataset provides the page-level transcriptions for 558 text pages written in three Indic scripts. Page-level text recognizers require excessive amount of data in comparison to word-level recognizers. On the IAM and READ datasets the state-of-the-art

Table 1. Publicly available handwritten datasets for Indic scripts and comparison to the dataset introduced in this work. All the datasets provide word-level transcriptions, except for the PBOK dataset.

Name	Script	#Writers	#Word Instances	#Lexicon
PBOK [1]	Bengali	199	21K	925
	Kannada	57	29K	889
	Odia	140	27K	1040
ROYDB [22]	Bengali	60	17K	525
	Devanagari	60	16K	1030
CMATERDB2.1 [3]	Bengali	300	18K	120
CENPARMI-U [23]	Urdu	51	19K	57
LAW [19]	Devanagari	10	27K	220
TAMIL-DB [28]	Tamil	50	25K	265
IIIT-HW-DEV [8]	Devanagari	12	95K	11,030
IIIT-HW-TELUGU [9]	Telugu	11	120K	12,945
IIIT-INDIC-HW-WORDS (This work)	Bengali	24	113K	11,295
	Gujarati	17	116K	10,963
	Gurumukhi	22	112K	11,093
	Kannada	11	103K	11,766
	Odia	10	101K	13,314
	Malayalam	27	116K	13,401
	Tamil	16	103K	13,292
	Urdu	8	100K	11,936
	8 scripts	**135**	**868K**	**97,060**

full-page text recognizer [31] has high error rates due to additional challenges like extremely skewed text, overlapping lines and inconsistent gaps between lines. Note that the IAM dataset is 3× bigger than PBOK dataset. Therefore, solving Indic HTR for full page text recognition requires a major effort and is beyond the scope of this work.

IIIT-HW-DEV [8] and IIIT-HW-TELUGU [9] are possibly the only Indic datasets that are comparable to the IAM dataset at word level. Both these datasets have around 100K word instances each with a lexicon size of over 10K unique words. In this work, we complement this effort and extend similar datasets in many other languages so as to cover 10 prominent Indic scripts. We introduce a unified database for handwritten datasets that are comparable to the IAM dataset in size and diversity. This dataset for Indic scripts is referred to as the IIIT-INDIC-HW-WORDS dataset. We compare our dataset to existing datasets in Table 1. We describe the dataset and discuss further details in the next section.

3 IIIT-INDIC-HW-WORDS

This section introduces the IIIT-INDIC-HW-WORDS dataset consisting of handwritten image instances in 8 different Indic scripts. The scripts present in the collection are Bengali, Gujarati, Gurumukhi, Kannada, Malayalam, Odia, Tamil

Table 2. Statistics of IIIT-INDIC-HW-WORDS dataset and IAM dataset.

Dataset	Script	Word Length (Avg)	#Instances		
			Train set	Val set	Test set
IAM	English	5	53,838	15,797	17,615
IIIT-INDIC-HW-WORDS	Bengali	7	82,554	12,947	17,574
	Gujarati	6	82,563	17,643	16,490
	Gurumukhi	5	81,042	13,627	17,947
	Kannada	9	73,517	13,752	15,730
	Odia	7	73,400	11,217	16,850
	Malayalam	11	85,270	11,878	19,635
	Tamil	9	75,736	11,597	16,184
	Urdu	5	71,207	13,906	15,517

and Urdu. Indian scripts belong to the Brahmic writing system. The scripts later evolved into two distinct linguistic groups: Indo-Aryan languages in the Northern India and Dravidian languages in the South India. Out of the 8 scripts discussed in this work, 5 scripts are of Indo-Aryan descent, and 3 are of Dravidian descent. Bengali, Gujarati, Gurumukhi, Odia, and Urdu belong to the Indo-Aryan family. Kannada, Malayalam, and Tamil are Dravidian languages. Indian scripts run from left to right, except for Urdu. Table 2 shows the statistics of this dataset. In the dataset, more than 100K handwritten word instances are provided for each script. The entire dataset is written by 135 natural writers aged between 18 and 70. More details about the dataset are mentioned on the dataset web-page.

Dutta et al. [9] propose an effective pipeline for the annotation of word-level datasets. We employ the same approach for collection and annotation generation for IIIT-INDIC-HW-WORDS. A large and appropriate vocabulary is selected to cover the language adequately. Participants are asked to write in a large coded space on an A4 paper. The written pages are scanned at 600 DPI using a flatbed scanner. Word images and associated annotations are automatically extracted from the scanned forms using image processing techniques. The average image height and width in the dataset is 288 and 1120. Samples that are wrongly segmented or those containing printed text, QR codes, and box borders are eliminated from the dataset. Such samples accounted for 4% of the total extracted samples. Except for removing wrongly segmented words, we do not perform any other kind of image pre-processing. Fig. 1 shows word instances for each of these scripts. We describe some of our observations here. The words found in Dravidian languages are longer than those present in Indo-Aryan languages. We plot the differences in word lengths between these two groups in Fig. 2. We also show the inter-class and intra-class variability in handwriting styles across Indic scripts in Fig. 1. The variable handwriting styles, noisy backgrounds, and choice of the unwanted pen makes it a diverse and challenging collection. Many samples in the dataset contain overwritten characters or poorly visible characters due to pen use. The varying background noise and paper quality also adds to the complexity of the dataset.

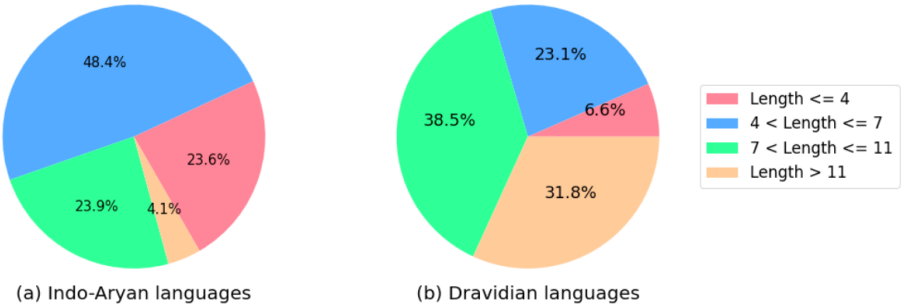
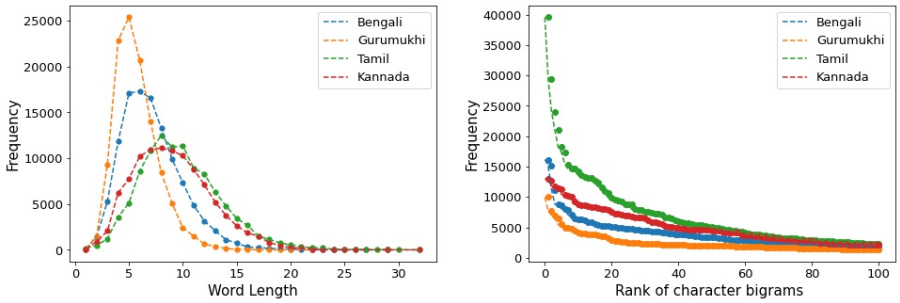


Fig. 2. Plots showing the variation in distribution of word lengths in IIIT-INDIC-HW-WORDS dataset for two linguistic groups: Indo-Aryan languages and Dravidian languages.



(a) Word length distribution (b) Character bigram distribution

Fig. 3. Distribution plots for four scripts: Bengali, Gurumukhi, Tamil, and Kannada. We observe that above distributions for Gujarati, Odia are similar to that of Bengali script. Distributions for Urdu and Malayalam script are comparable to those of Gurumukhi and Tamil script respectively.

The text lexicon for the datasets is sampled from the Leipzig corpora collection [11]. It consists of text files with content from newspaper articles, Wikipedia links, and random websites. More than 10K unique words are sampled from these collections in each language to generate the coded pages for participants to write. The number of unique characters per script includes the basic characters in the respective script's Unicode block and some special characters. The statistics of lexicon size for the IIIT-INDIC-HW-WORDS collection are listed in Table 1. Fig. 3a shows the distribution of word length across different datasets. The plot 3b shows that the character bigrams follow Zipf's law. We observe that the law holds true for character n-grams as well with $n = 3, 4, 5, 6$.

The annotated dataset consists of image files along with a text file containing the corresponding label information. The text labels are encoded using Unicode. We also release the train, test, and validation splits for the word recognition task.

Around 70% of the instances are added to the training set, 15% to the test set, and the remaining instances comprise the validation set. We ensure that 9-12% of the test labels are out-of-vocabulary(OOV) words. Test sets with a high rate of OOV samples are challenging. The evaluated metrics on such sets inform whether a proposed solution is biased towards the vocabulary of training and validation sets. The total number of samples having out-of-vocabulary text labels in the test sets varies from 35% to 40% per script.

4 Baseline for Text Recognition

Unconstrained offline handwritten text recognition (HTR) is the task of identifying the written characters in an image without using any external dictionaries for a language. Previous works in text recognition [26,2,4,21] show that deep neural networks are very good at solving this problem due to their generalization capability and representational power. Several deep neural network architectures have been proposed for recognition of scene text, printed text, handwritten content. Baek et al. [2] propose a four-stage text recognition framework derived from existing scene text recognition architectures. The recognition flow at different stages of the framework is demonstrated in Fig. 4. This framework can be applied to the HTR task as well. In this section, we discuss and study the different stages of the pipeline. We identify an appropriate architecture for HTR and establish a baseline on the IIIT-INDIC-HW-WORDS dataset.

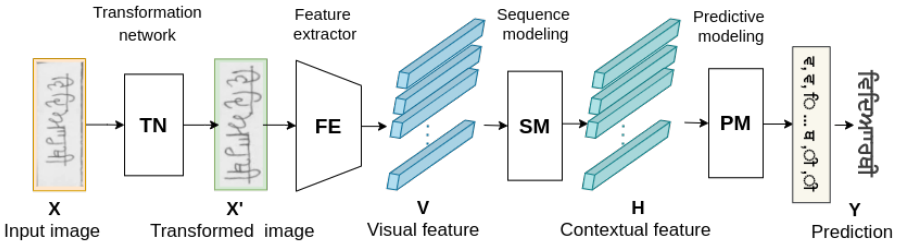


Fig. 4. Generic pipeline for text recognition. Here, we demonstrate the flow of a sample image **X** through the pipeline to generate a text prediction **Y**. The sample shown here is written in Gurumukhi script.

Transformation Network(TN): Diverse styles observed in handwriting data are a significant challenge for HTR. A transformation block learns to apply input-specific geometric transformations such that the end goal of text recognition is simplified. In other words, this module reduces the burden of the later stages of the pipeline. Spatial Transformer Network (STN) [17] and its variants are commonly used to rectify the input images. In this work, we experiment with two

types of rectification, affine transformation (ATN), and thin-plate spline transformation(TPS). Affine module applies a transformation to rectify the scale, translation, and shear. TPS applies non-rigid transformation by identifying a set of fiducial points along the upper and the bottom edges of the word region.

Feature Extractor (FE): The transformed image is forwarded to a convolutional neural network(CNN) followed by a map-to-sequence operation to extract feature maps. This visual feature sequence V has a corresponding distinguishable receptive field along the horizontal line of the input image. Words are recognized by predicting the characters at every step of the feature sequence V. We study two commonly used CNN architecture styles: VGG [27] and RESNET [15]. VGG-style architecture comprises multiple convolutional layers followed by a few fully connected layers. RESNET-style architecture uses residual connections and eases the training of deep CNNs.

Shi et al. [26] tweak the original VGG architecture so that the generated feature maps have larger width to accommodate for recognition of longer words. Dutta et al. [8] introduce a deeper architecture with residual connections. We study both these architectures in this work. We refer to these as HW–VGG and HW–RESNET.

Sequence Modeling(SM): The computed visual sequence V lacks contextual information, which is necessary to recognize characters across the sequence. Therefore, the feature sequence is forwarded to a stack of recurrent neural networks to capture the contextual information. Bidirectional LSTM(BLSTM) is the preferred block to compute the new feature sequence H as it enables context modeling from both directions. Due to its success [26,21,7,2], we use a 2 layer BLSTM architecture with 256 hidden neurons in each layer as the SM module in our experiments.

Predictive Modeling(PM): This module is responsible for decoding a character sequence from the contextual feature H. To decode and recognize the characters, this block learns the alignment between the feature sequence H and the target character sequence. One of the commonly used methods to achieve this is Connectionist Temporal Classification(CTC) [13]. It works by predicting a character for every frame in the sequence and removing recurring characters and blanks.

Data Augmentation: Training deep networks to learn generalized features is crucial for the task of handwriting recognition. Work done in [7,9,29] shows that data augmentation can improve HTR performance on Latin and Indic datasets. It enables the architecture to learn invariant features for the given task and prevents the networks from overfitting. In this work, we apply affine and elastic transformations. This is done to imitate the natural distortions and variations observed in handwriting data. We also apply brightness and contrast augmentation to learn invariant features for text and background.

Table 3. Results on Bengali and Tamil script in IIIT-INDIC-HW-WORDS dataset for common HTR architectures. TN-FE-SM-PM refers to the specific modules used for text recognition. BLSTM-CTC is used for SM-PM stages in the experiments.

#	TN	FE	CER		Params
			Bengali	Tamil	x10^6
M1	None	HW–VGG	11.48	7.25	8.35
M2	ATN	HW–VGG	9.41	5.99	8.38
M3	ATN	HW–RESNET	5.47	1.47	16.54
M4	TPS	HW–RESNET	5.22	1.38	17.15

Results: In this section, we evaluate and discuss four architecture combinations in the HTR pipeline. Through this setup, we aim to determine the best model for the Indic HTR task. The model architectures are listed in Table 3. For every upgrade done from M1 to M4, we introduce better alternatives in a single stage of the HTR pipeline. We observe the improvement introduced by a specific module. The best performing architecture is identified from the combinations while considering the trade-off between evaluation metrics and computational complexity. We evaluate the performance of these architectures on two datasets: Tamil and Bengali. The Tamil dataset belongs to the Dravidian linguistic group, and the Bengali dataset belongs to the Indo-Aryan group.

We compare the alternatives in TN and FE stages of the HTR pipeline against character error rate(CER) and total number of parameters associated with the architecture. The results obtained are presented in Table 3. For feature extraction stage, replacing HW–VGG with HW–RESNET architecture(M2 $\rightarrow$ M3) increases the number of parameters by 2×. However, architecture M3 significantly improves the error rate by at least 4%. As the number of layers are increased, architecture M3 benefits from the network's increased complexity and representation power.

Changing the transformation network from ATN to TPS(M3 $\rightarrow$ M4) reduces the error rate by minor margin only. We conduct another experiment to understand the effectiveness of TN. For this study, we train two architectures with and without TN for different sizes of training data. We train the architectures with 10K, 30K, 50K, and 82K samples for Bengali script. The remaining stages of the pipeline are chosen as HW–RESNET-BLSTM-CTC. The comparison plot in Fig. 5 shows that the WER for the TPS variant reduces significantly when the training datasets have limited samples. With limited training data, the TN stage is crucial to build robust HTR. The introduced changes in TN and FE stages increases the architecture's complexity resulting in smaller error rates.

With the HTR pipeline as TPS-HW–RESNET-BLSTM-CTC, we train and evaluate its performance on all the scripts in the IIIT-INDIC-HW-WORDS dataset. The samples are augmented randomly with affine, elastic, and color transformations. The model is fine-tuned on the pre-trained weights from the IAM dataset. To this end, we train the architecture on IAM dataset as well and the WER and CER obtained are 13.17 and 5.03 respectively. The obtained performance metrics are reported in Table 4. We report error rates on two sets; the test set and its

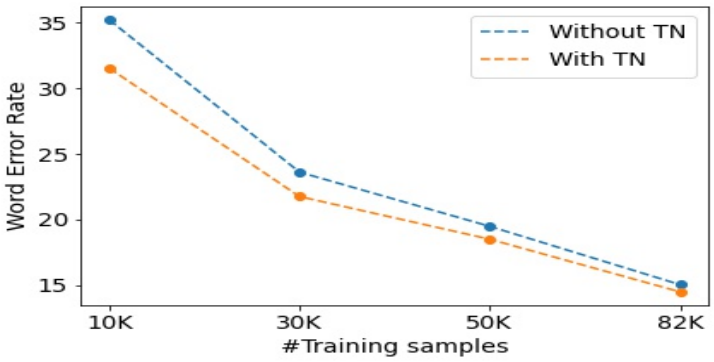


Fig. 5. Word error rate(WER) vs. training size for two architectures: with TPS TN and without TN. The FE, SM, PM stages are fixed as HW–RESNET, BLSTM, CTC. The results are shown for Bengali script in IIIT-INDIC-HW-WORDS dataset.

Table 4. Results on IIIT-INDIC-HW-WORDS dataset. Metrics listed are computed on the test set and out-of-vocabulary(OOV) test set. CER is character error rate and WER is word error rate. Characters refers to number of unique Unicode symbols in each dataset.

Script	Characters	Test		OOV Test	
		CER	WER	CER	WER
Bengali	91	4.85	14.77	3.71	16.65
Gujarati	79	2.39	11.39	5.27	25.8
Gurumukhi	84	3.42	12.78	6.85	23.97
Odia	81	3.00	14.97	4.97	23.40
Kannada	83	1.03	5.90	1.59	9.39
Malayalam	95	1.92	9.85	2.06	10.6
Tamil	72	1.28	7.38	1.88	8.25
Urdu	81	3.67	15.00	9.33	33.23

subset containing only OOV words. From the table, the error rates observed for Kannada script is the lowest and the error rates for Urdu script are the highest. It is interesting to observe similar performances within the Indo-Aryan group and Dravidian group. We also note that the Urdu dataset is more challenging than other datasets due to high error rates, despite having fewer writing styles. Fig. 6 shows the predicted text for selective samples from the IIIT-INDIC-HW-WORDS dataset. The model fails to recognize the samples presented in the third column by less than two characters. The recognized text for these samples is incorrect due to confusing character pairs or predicting an extra character at the end of the text string. For the samples presented in the fourth column, the longer length of words causes the recognizer to make mistakes and generate shorter predictions.

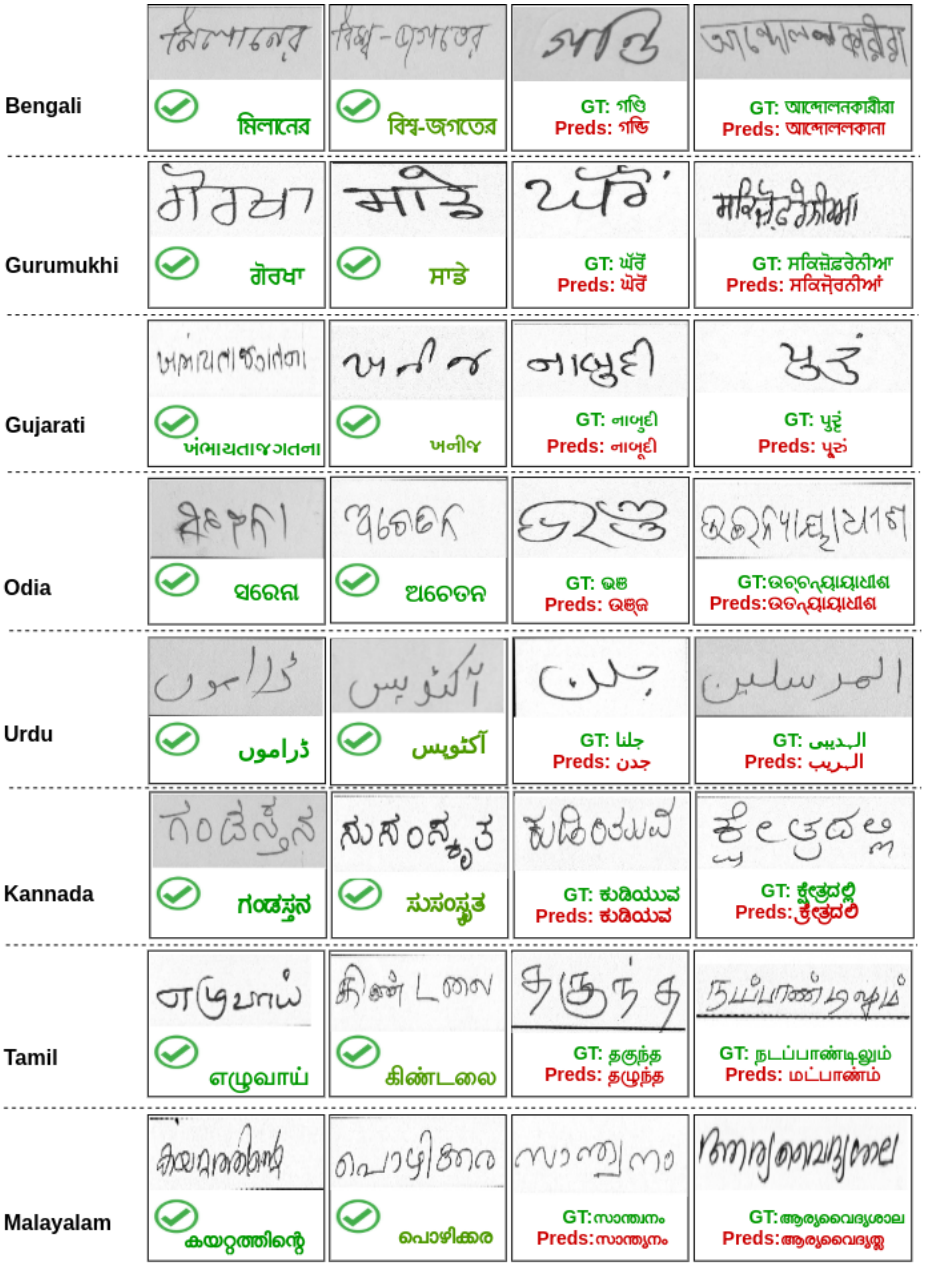


Fig. 6. Qualitative results on the IIIT-INDIC-HW-WORDS dataset. Word images are converted to grayscale and forwarded to the recognizer. Column 1 & 2 present correctly predicted samples. Column 3 & 4 shows incorrect predictions. GT and Preds refer to ground truth and predicted label respectively.

5 Analysis

Training deep neural architectures for a specific task requires optimization of millions of parameters. Due to their large size, training from randomly initialized weights requires a lot of time and hyper-parameter tuning. Pre-trained weights computed for other tasks are generally used to initialize new networks [6,30]. This is also referred to as transfer learning. This technique is especially meaningful in HTR where the number of training samples is limited in a specific domain or script [18,12]. Given a dataset with limited training samples, performing transfer learning from a large dataset eases the training and improves the performance.

Table 5. Effect of pre-training with varying training size. The results shown here are computed on the IIIT-INDIC-HW-WORDS Bengali dataset.

Training size	Pre-trained Weights	WER	CER
10K	None	31.54	9.64
	IAM	29.28	8.98
	IIIT-HW-DEV	28.73	8.94
	IIIT-HW-TELUGU	**28.79**	**8.71**
30K	None	21.76	6.88
	IAM	20.37	6.43
	IIIT-HW-DEV	**20.10**	**6.36**
	IIIT-HW-TELUGU	20.35	6.38
50K	None	18.08	5.67
	IAM	**17.23**	**5.49**
	IIIT-HW-DEV	18.30	5.72
	IIIT-HW-TELUGU	17.8	5.67
82K	None	15.34	4.97
	IAM	**14.77**	**4.85**
	IIIT-HW-DEV	15.35	4.97
	IIIT-HW-TELUGU	15.0	5.04

For HTR domain, Bluche et al. [4] discuss the advantages of training recognizers using big multi-lingual datasets in Latin scripts. Dutta et al. [9] show that using pre-trained architectures for fine tuning on Indic datasets improves the performance. They utilize weight parameters learnt on the IAM dataset to reduce the error rate on the IIIT-HW-DEV and the IIIT-HW-TELUGU datasets. Whereas, recent research [16] points out a network trained from scratch for long intervals is not worse than a network finetuned on pre-trained weights for detection and segmentation task. In this study, we explore whether pre-training is useful within the context of handwritten data across multiple scripts. We use three scripts to conduct these experiments: English, Devanagari and Telugu. We also explore the utility of pretrained architectures with varying number of training samples.

In this study, we explore the training architectures using randomly initialized weights, and pre-trained weights from IAM, IIIT-HW-DEV and IIIT-HW-TELUGU

datasets. We also explore these pre-training strategies for varying sizes of training data. We present the results on the Bengali data in the IIIT-INDIC-HW-WORDS dataset. The networks are trained for same number of iterations using Adadelta optimization method. The final M4 architecture is trained with data augmentation in this experiment and the results obtained are presented in Table 5. The benefit of using pre-trained architectures for smaller training datasets is notable and gives an improvement of 2.75% in word error rate(WER). As the available training data increases, the reduction in error rates is not apparent. We observe that a randomly initialized network is not necessarily worse at recognition than the network using pre-trained weights for initialization.

Through the above experiment, we also conclude that language similarity is not a major contributing factor to performance improvement for HTR task. The datasets used in the study are written in Latin, Devanagari, and Telugu script. The corresponding datasets for these scripts are IAM, IIIT-HW-DEV, IIIT-HW-TELUGU. Bengali and Devanagari scripts share similarities and both the scripts belong to the Indo-Aryan group. However, results show that pre-training from similar language outperforms other pre-training techniques only for one of the cases and also by very small margin. Interestingly, the pretrained weights from the IAM dataset provide best results as the training data increases to 82K samples.

From this brief study, we conclude that pre-training is especially meaningful for handwritten tasks when the available training data is limited. Therefore, pre-trained architectures can be utilized to build recognizer for historical data or regional scripts. Transcriptions for a few samples are sufficient to fine-tune the recognizers for such target tasks. We also observe that pretraining from similar scripts does not supplement the training process for Indic scripts.

6 Summary

In this work, we extend our earlier efforts in creating Devanagari [8] and Telugu [9] datasets. We introduce a collective handwritten dataset for 8 new Indian scripts: Bengali, Gujarati, Gurumukhi, Kannada, Malayalam, Odia, Tamil, and Urdu. We hope the scale and the diversity of our dataset in all 10 prominent Indic scripts will encourage research on enhancing and building robust HTRs for Indian languages. It is essential to continue improving Indic datasets to enable Indic HTR development, and therefore, future work will include enriching the dataset with more writers and natural variations. Another important direction is to create handwritten data at line, paragraph and page level.

For the introduced dataset, we establish a high baseline on the 8 scripts present in the IIIT-INDIC-HW-WORDS dataset and discuss the effectiveness of the HTR modules. We also conduct a brief study to explore the utility of pre-training recognizers on other scripts.

Acknowledgments

The authors would like to acknowledge the funding support received through IMPRINT project, Govt. of India to accomplish this project. We would like to thank all the volunteers who contributed to this dataset. We would also like to express our gratitude to Aradhana, Mahender, Rohitha and annotation team for their support in data collection. We also thank the reviewers for their detailed comments.

References

1. Alaei, A., Pal, U., Nagabhushan, P.: Dataset and Ground Truth for Handwritten Text in Four Different Scripts. IJPRAI (2012) 2, 1
2. Baek, J., Kim, G., Lee, J., Park, S., Han, D., Yun, S., Oh, S.J., Lee, H.: What Is Wrong With Scene Text Recognition Model Comparisons? Dataset and Model Analysis. In: ICCV (2019) 4, 4
3. Bhowmik, S., Malakar, S., Sarkar, R., Basu, S., Kundu, M., Nasipuri, M.: Off-line Bangla handwritten word recognition: a holistic approach. Neural Computing and Applications (2019) 2, 1
4. Bluche, T., Messina, R.: Gated convolutional recurrent neural networks for multilingual handwriting recognition. In: ICDAR (2017) 1, 4, 5
5. Bluche, T., Ney, H., Kermorvant, C.: Feature extraction with convolutional neural networks for handwritten word recognition. In: ICDAR (2013) 1
6. Donahue, J., Jia, Y., Vinyals, O., Hoffman, J., Zhang, N., Tzeng, E., Darrell, T.: Decaf: A deep convolutional activation feature for generic visual recognition. In: ICML (2014) 5
7. Dutta, K., Krishnan, P., Mathew, M., Jawahar, C.V.: Improving CNN-RNN hybrid networks for handwriting recognition. In: ICFHR (2018) 4, 4
8. Dutta, K., Krishnan, P., Mathew, M., Jawahar, C.: Offline Handwriting Recognition on Devanagari Using a New Benchmark Dataset. In: DAS (2018) 1, 1, 2, 4, 6
9. Dutta, K., Krishnan, P., Mathew, M., Jawahar, C.: Towards Spotting and Recognition of Handwritten Words in Indic Scripts. In: ICFHR (2018) 1, 1, 2, 3, 4, 5, 6
10. Fischer, A., Keller, A., Frinken, V., Bunke, H.: Lexicon-free handwritten word spotting using character HMMs. PR (2012) 1, 2
11. Goldhahn, D., Eckart, T., Quasthoff, U.: Building Large Monolingual Dictionaries at the Leipzig Corpora Collection: From 100 to 200 Languages. In: LREC (2012) 3
12. Granet, A., Morin, E., Mouchère, H., Quiniou, S., Viard-Gaudin, C.: Transfer Learning for Handwriting Recognition on Historical Documents. In: ICPRAM (2018) 5
13. Graves, A., Fernández, S., Gomez, F., Schmidhuber, J.: Connectionist Temporal Classification: Labelling Unsegmented Sequence Data with Recurrent Neural Networks. In: ICML (2006) 4
14. Grosicki, E., El-Abed, H.: ICDAR 2011-French Handwriting Recognition Competition. In: ICDAR (2011) 1
15. He, K., Zhang, X., Ren, S., Sun, J.: Deep Residual Learning for Image Recognition. In: CVPR (2016) 1, 4

16. He, K., Girshick, R., Dollár, P.: Rethinking Imagenet Pre-training. In: CVPR (2019) 5
17. Jaderberg, M., Simonyan, K., Zisserman, A., Kavukcuoglu, K.: Spatial transformer networks. In: NIPS (2015) 4
18. Jaramillo, J.C.A., Murillo-Fuentes, J.J., Olmos, P.M.: Boosting handwriting text recognition in small databases with transfer learning. In: ICFHR (2018) 5
19. Jayadevan, R., Kolhe, S.R., Patil, P.M., Pal, U.: Database Development and Recognition of Handwritten Devanagari Legal Amount Words. In: ICDAR (2011) 2, 1
20. Marti, U.V., Bunke, H.: The IAM-database: an English sentence database for offline handwriting recognition. IJDAR (2002) 1, 2
21. Puigcerver, J.: Are multidimensional recurrent layers really necessary for handwritten text recognition? In: ICDAR (2017) 1, 4, 4
22. Roy, P.P., Bhunia, A.K., Das, A., Dey, P., Pal, U.: HMM-based Indic handwritten word recognition using zone segmentation. Pattern recognition (2016) 2, 1
23. Sagheer, M.W., He, C.L., Nobile, N., Suen, C.Y.: A New Large Urdu Database for Off-Line Handwriting Recognition. In: ICIAP (2009) 1
24. Sánchez, J.A., Romero, V., Toselli, A.H., Vidal, E.: ICFHR2014 competition on handwritten text recognition on transcriptorium datasets (HTRtS). In: ICFHR (2014) 2
25. Sanchez, J.A., Romero, V., Toselli, A.H., Villegas, M., Vidal, E.: ICDAR2017 Competition on Handwritten Text Recognition on the READ Dataset. In: ICDAR (2017) 2
26. Shi, B., Bai, X., Yao, C.: An End-to-End Trainable Neural Network for Image-Based Sequence Recognition and Its Application to Scene Text Recognition. PAMI (2016) 4, 4, 4
27. Simonyan, K., Zisserman, A.: Very Deep Convolutional Networks for Large-Scale Image Recognition. In: ICLR (2015) 1, 4
28. Thadchanamoorthy, S., Kodikara, N., Premaretne, H., Pal, U., Kimura, F.: Tamil Handwritten City Name Database Development and Recognition for Postal Automation. In: ICDAR (2013) 2, 1
29. Wigington, C., Stewart, S., Davis, B., Barrett, B., Price, B., Cohen, S.: Data Augmentation for Recognition of Handwritten Words and Lines Using a CNN-LSTM Network. In: ICDAR (2017) 4
30. Yosinski, J., Clune, J., Bengio, Y., Lipson, H.: How transferable are features in deep neural networks? NIPS (2014) 5
31. Yousef, M., Bishop, T.E.: OrigamiNet: Weakly-Supervised, Segmentation-Free, One-Step, Full Page Text Recognition by learning to unfold. In: CVPR (2020) 2