2024 1st International Conference on Cognitive, Green and Ubiquitous Computing (IC-CGU) | 979-8-3503-6178-0/24/$31.00 ©2024 IEEE | DOI: 10.1109/IC-CGU58078.2024.10530719

SYNTHETIC DATA GENERATION FOR DOCUMENT TEXT RECOGNITION

Ashlin Deepa R N
Computer Science and Engineering
Gokaraju Rangaraju Institute of Engineering and Technology
Hyderabad, India
rndeepa.pradeep@gmail.com

Boda Srija
Computer Science and Engineering
Gokaraju Rangaraju Institute of Engineering and Technology
Hyderabad, India
srijaboda5@gmail.com

Maria Jabeen
Computer Science and Engineering
Gokaraju Rangaraju Institute of Engineering and Technology
Hyderabad, India
jabeen.maria2003@gmail.com

Kushi Reddy Kankar
Computer Science and Engineering
Gokaraju Rangaraju Institute of Engineering and Technology
Hyderabad, India
kushireddykankar@gmail.com

Bollepalli Jhansi Lakshmi
Computer Science and Engineering
Gokaraju Rangaraju Institute of Engineering and Technology
Hyderabad, India
jhansibollepalli18@gmail.com

Y.Vijayalata
Computer Science and Engineering
Vardhaman College of Engineering
Hyderabad, India
vijaya@ieee.org

***Abstract*—Handwritten text recognition software is used to recognize and extract text from scanned documents. The fundamental goal of this technology is to transform printed or handwritten text into an easily readable electronic format. The numerous characters and enormous quantity of the information for Indian languages, however, causes explicit preprocessing to take time. This necessity for explicit preprocessing has been replaced by synthetic data creation techniques, which overcome several difficulties and speed up the process. Artificially produced data that closely mimics actual observations is referred to as synthetic data. In situations where getting real data is difficult or expensive, it offers a workable substitute for training machine learning models. In this work, we propose a data preprocessing technique that creates synthetic data files from the already-existing collection of Indian languages. We use a pre-trained language model called FastText model, which is capable of creating word embeddings, to generate synthetic dataset from the real-time dataset. With the aid of the generated synthetic datasets, document text recognition systems can undergo extensive training and testing, allowing them to grasp the complexities of Indian languages and perform accurate text extraction from scanned documents.**

***Index Terms*—Indic handwritten text, Synthetic data, STN, BiLSTM Network, word embeddings, FastText, pre-trained language model**

I. Introduction

Handwritten Text Recognition (HTR) is a field of artificial intelligence and computer vision that aims to convert handwritten text into machine-readable digital text. With the vast linguistic diversity in India, HTR plays a crucial role in preserving, digitising, and making accessible the rich cultural and historical heritage stored in handwritten documents and manuscripts. The languages often rely on complex word formations, involving ligatures, conjuncts, and stacked characters. As a result, developing an accurate and robust HTR system for Indian languages poses several significant challenges.[20] by Vijay proposed an online telugu character level handwritten text recognizer. Saini created the most extensive and publicly accessible synthetic OCR benchmark dataset for Indic languages[12].

Synthetic data generation allows the creation of custom datasets tailored to particular HTR applications, such as historical manuscripts or medical records. It allows for the augmentation of existing datasets, increasing their size and diversity. Synthetic data generation provides an efficient means of creating labelled data in a controlled environment, reducing the need for extensive manual labelling, [13] demonstrates the usefulness of Synthetic Data. Synthetic data can help in pre-training HTR models on a larger dataset before fine-tuning on a smaller real-world dataset. It reduces the cost and time required for preprocessing, thus enabling more efficient and accurate recognition of handwritten text in Indic languages.

A pre-trained language model called FastText[19] is used in our work. It is a powerful and efficient library developed by Facebook AI Research for text classification and representation learning tasks. It provides a range of functions and capabilities that enable users to effectively work with textual data such as text categorization, and word vector representations.

This paper introduces an innovative method for generating synthetic data to enhance document text recognition, a critical aspect within the fields of machine learning and natural language processing. Our methodology involves the creation of artificial samples that closely mimic real-world documents, thereby enriching training datasets and bolstering the robustness of text recognition models.

TABLE I
EXISTING PAPERS ON HTR AND OCR

S.no	Author	Dataset - Languages used	Dataset size	Technique	Description
1.	Kang[14]	IAM, GW, CVL - English, French, and Catalan	1,539 handwritten pages comprising 115,320 words.	Used CNN, RNN, and VGG-19-BN Architecture.	Proposed a novel unsupervised writer adaptation application for handwritten text recognition. Generic HTR model, trained only with synthetic data.
2.	Varga[15]	IAM - English	657 writers generated 13,353 images containing handwritten lines of text.	Hidden Markov Model (HMM) based cursive handwritten textline recognizer.	A perturbation model for generating synthetic textlines from existing cursive handwritten textlines.
3.	Roy[9]	CMATER - Devanagiri, Bangla	32x32 RGB colored, 6000 images	PCA, HOG, PHOG	A novel approach to generate a synthetic dataset for handwritten word recognition systems
4.	Dutta[16]	RoyDB - Devanagiri	95,381 word samples	CNN-RNN Hybrid Architecture	Benchmarking a new dataset using a CNN-RNN hybrid network. Fine-tuning the network using synthetic and real handwritten data.
5.	Esma[17]	UNIPEN, ET - Turkish	61,351 lowercase and 28,069 uppercase characters.	CNN-BLSTM network	Used synthetic data to develop an online handwriting recognition system for Turkish.
6.	Johannes[18]	IAM, ICFHR2016 READ - English, German	400 pages	Encoder-decoder framework using CNN, RNN, BiLSTM	HTR task with an attention-based Seq2Seq architecture using deep CNN-RNN encoder and RNN decoder.

II. Related Works

Image data generation involves the creation of artificial images to augment existing datasets, thereby enhancing model performance and robustness. Principal Component Analysis (PCA) is a powerful dimensionality reduction technique commonly employed in image data generation to extract essential features and reduce the computational complexity of high-dimensional image datasets. In [5] by Turk, PCA was introduced to extract key facial features. [1] by Krizhevsky introduced data augmentation techniques that reduces overfitting. Unlike PCA, VAEs can capture complex non-linear relationships by introducing probabilistic encoders and decoders. The encoder maps data into a latent space distribution, and the decoder generates data from samples in this distribution.

Autoencoders learn to encode and then reconstruct images, enabling the creation of synthetic images by modifying the encoded representations. Variational Autoencoders (VAEs) offer a probabilistic approach to data generation. The paper [2] by Kingma introduced VAEs, demonstrating their ability to generate diverse and meaningful images while encoding underlying data distributions. The latent space in VAEs is often continuous and has a well-defined probabilistic interpretation. GANs(Generative Adversarial Networks) do not explicitly model the latent space in the same way VAEs do. The latent space in GANs is implicitly learned through the generator's architecture, but it may not have a direct probabilistic interpretation.

Generative Adversarial Networks (GANs) are cutting-edge deep learning models which consist of two neural networks: a generator and a discriminator. The generator generates synthetic images while the discriminator tries to distinguish between real and fake images. In [3] by Goodfellow, GANs were introduced, producing astonishingly realistic images. [4] by Karras, showcasing their prowess in high-resolution image synthesis. [6] by Xie introduced a method that leverages unlabeled data, showcasing the potential of self-supervised techniques. Additionally, [7] by Zhang proposed mixing pairs of images and labels, yielding more robust models.

Synthetic data can effectively represent multiple languages and scripts, enabling the training of models that recognize and understand diverse linguistic content. Synthetic images are often created by applying transformations like rotation, scaling, and distortion to existing images. [8] by Praveen Krishnan proposes a framework to render large-scale synthetic data for handwritten images. [9] by Roy shows experiments conducted using synthetic data to improve the recognition accuracy of isolated characters and words.

Synthetic data offers multiple advantages such as circumventing privacy concerns, alleviating data scarcity issues. It can be tailored to specific use cases, enhancing model performance in targeted tasks and can also simulate challenging conditions such as degraded or distorted text. It is a cost-effective alternative to manually collecting and annotating large volumes of real-world data.

III. Proposed Methodology

Synthetic data generation for document text recognition involves creating artificial samples that mimic real-world documents, aiding in training robust models by increasing diversity and quantity of training data. We've successfully produced synthetic data for document text recognition in four languages: Kannada, Tamil, Telugu, and Malayalam.

The Fig. 1 illustration offers an overview of the structure of our proposed model.

The proposed model consists of two stages: (i) Encoding of images (ii) FastText model for Synthetic data generation

A. Encoding of Images

In our image encoding pipeline, we have combined Spatial Transformer Networks (STN) and Bidirectional Long Short-Term Memory (BiLSTM) networks. This collaborative approach yields a dynamic feature sequence by effectively

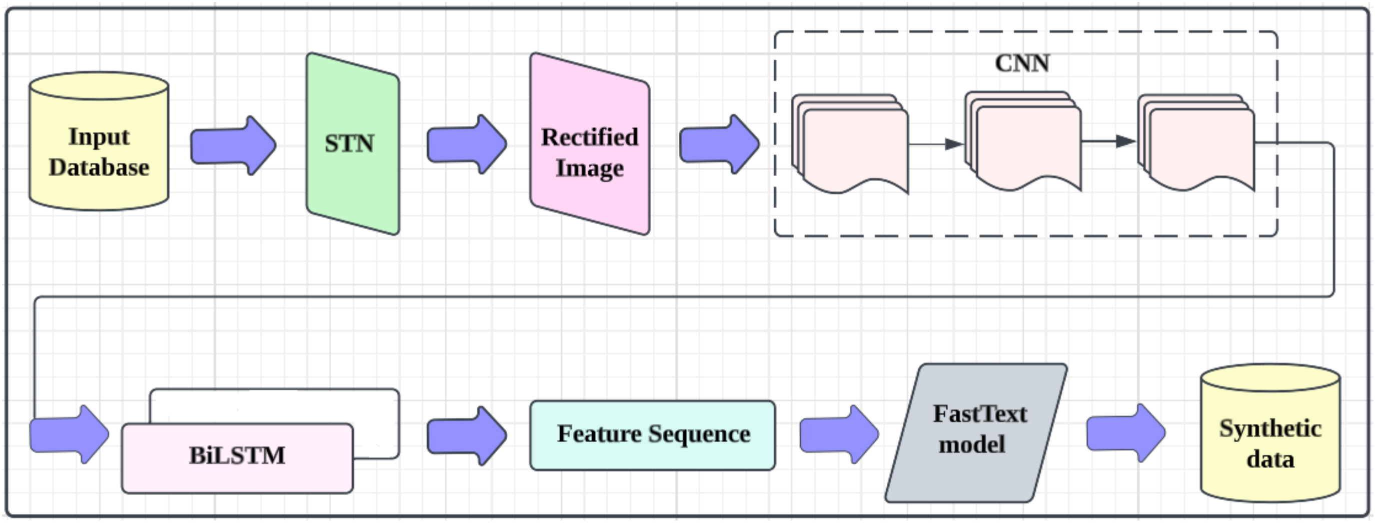


Fig. 1. Overall Architecture of synthetic data generation

capturing spatial transformations and contextual dependencies within the images.

The STN component orchestrates precise geometric adjustments, allowing the neural network to focus on salient image regions while accommodating variations in scale, rotation, and perspective. Concurrently, the BiLSTM network engages bidirectional processing, intelligently capturing intricate patterns and relationships within the transformed image data. This synergy produces a refined and comprehensive feature sequence, primed for further analysis.

1) Spatial Transformer Network: The Indian language dataset we've been working with has some intriguing characteristics, including rotated and twisted elements that require careful rectification before we can proceed with generating synthetic data. To tackle this particular challenge, we've incorporated the use of Spatial Transformer Networks (STN). STN[21] is a sophisticated technology that excels at correcting the spatial orientation of these images, effectively producing a set of meticulously rectified visuals. Through the integration of STN into our process, we not only address the distortion issue but also elevate the overall quality and precision of the resulting images.

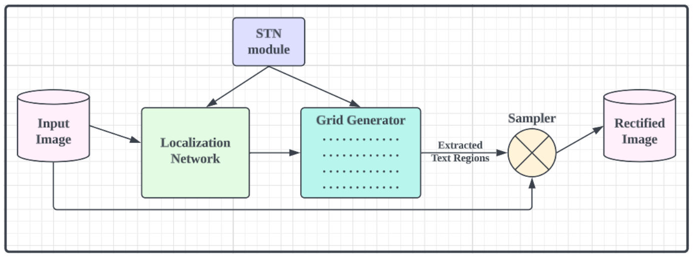


Fig. 2. Spatial Transformer Network Architecture

The STN is fundamentally comprised of three essential components: the localization network, the grid generator, and the sampler. The localization network predicts transformation parameters, such as translation, rotation, and scaling, based on the input data. These parameters are then used by the grid generator to create a sampling grid that defines how the input should be transformed. Finally, the sampler applies the grid to the input image, effectively warping it according to the predicted transformation.

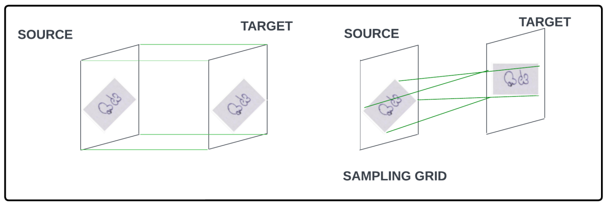


Fig. 3. STN Example

2) Bidirectional Long Short-Term Memory: The rectified images, which STN has skillfully fine-tuned, seamlessly flow into a CNN-BiLSTM model. This BiLSTM model extracts a sequence of features from the images, capturing their intricate nuances and patterns. This sequence of features is then effortlessly passed along to the FastText model, a versatile tool highly regarded for its prowess in natural language processing tasks. In the realm of our process, the FastText model ingeniously harnesses these extracted features to craft a compelling synthesis of data, ultimately yielding a diverse and authentic collection of synthetic content.

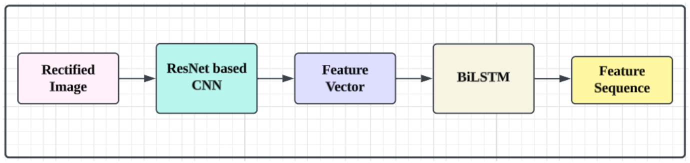


Fig. 4. CNN-Bidirectional Long Short-Term Memory Architecture

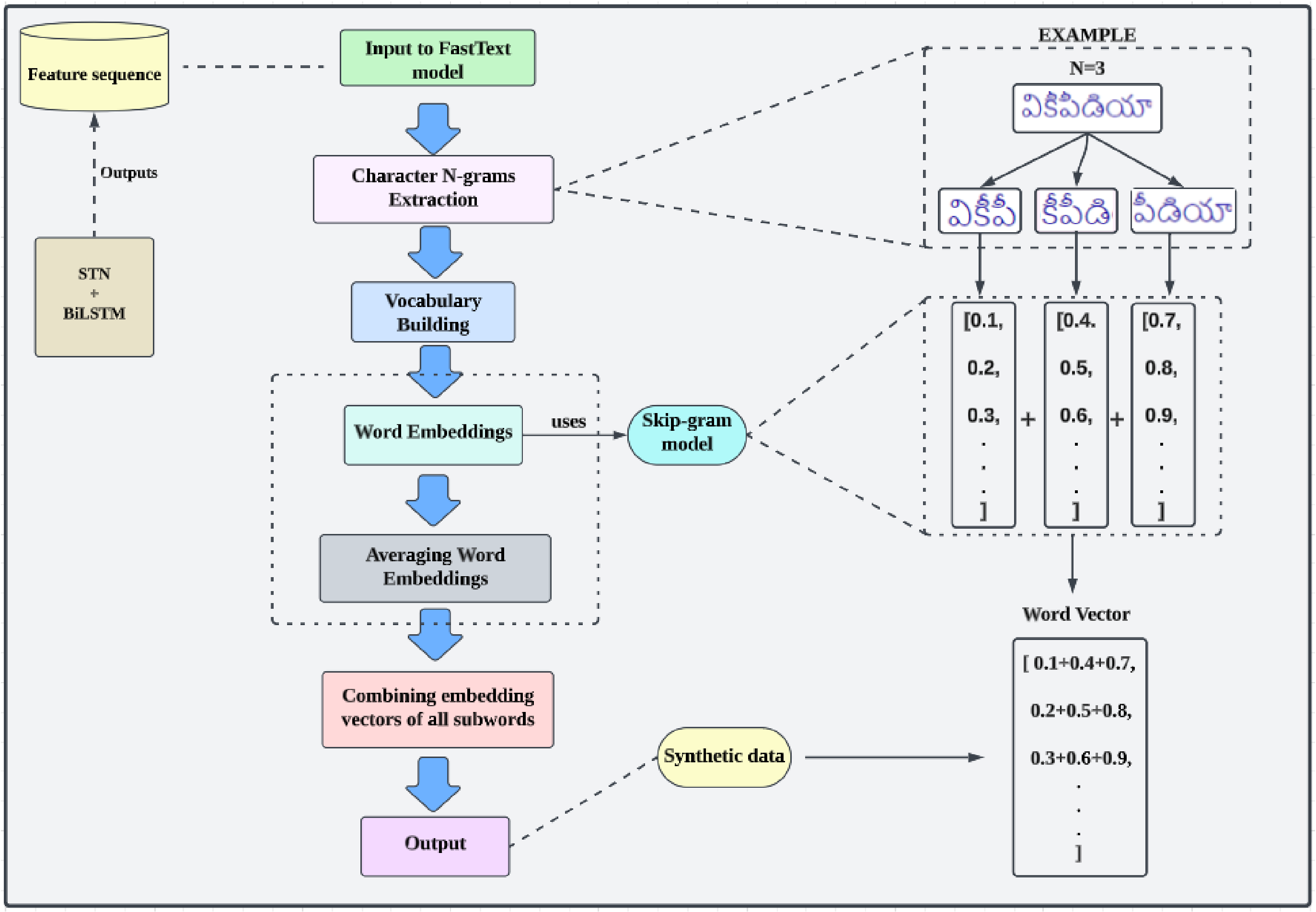


Fig. 5. FastText model

B. *FastText model for Synthetic data generation*

FastText is a highly regarded natural language processing (NLP) model developed by Facebook. It excels at processing text data, which is typically provided in the form of sentences or documents. What sets FastText apart is its unique approach to word representation. Instead of considering words as atomic units, it breaks them down into smaller subword units using character n-grams. These subword units capture morphological information and help in handling out-of-vocabulary words more effectively. Feature sequences resulting from using STN and BiLSTM are given as an input to the FastText model. Then, character n-gram extraction occurs. These n-grams capture local character patterns in the input text. The essence of the n-gram model, which FastText employs, lies in the idea that the probability of a word in a sentence can be estimated based on the probabilities of its preceding n-1 words. In the training process, FastText leverages n-grams to predict the succeeding word in a sentence by considering the context of previous words and subword units. A vocabulary using unique character n-grams and words is built. Using a variant of the Skip-gram model, FastText learns continuous word embeddings for each word in the vocabulary. It aims to predict the context (neighbouring words) given a target word or subword. Through this predictive approach, the model learns word and subword embeddings, which are distributed representations that capture the semantic meaning and contextual information of words. The averaged embedding vectors serve as synthetic data examples. A remarkable benefit of using subword representations is that the model gains the ability to comprehend the meaning and context of words it has never encountered before. In this way, synthetic data is generated using the FastText model. With the it's efficiency, accuracy, and versatility, FastText has become widely popular in the NLP community. Researchers and practitioners leverage the versatile capabilities of FastText for a range of tasks, including text classification, sentiment analysis, machine translation, and beyond. This makes FastText an essential and invaluable tool in contemporary natural language processing.

IV. RESULT

We utilised the openly accessible benchmark dataset, iiit-indichw-words[22], in our study. Our focus encompassed four significant Indian languages for synthesising data: Kannada, Malayalam, Tamil, and Telugu. The dimensions of the training, validation, and testing datasets for these languages are outlined in Table II.

TABLE II
DATASET SPLIT FOR DIFFERENT LANGUAGE SCRIPTS

Language Script	Train Set	Validation Set	Test Set
Kannada	73,517	13,752	15,730
Malayalam	85,270	11,878	19,635
Tamil	75,736	11,597	16,184
Telugu	80,694	20,049	17,911

The iiit-indichw-words[22] dataset captures images at the word level, exhibiting variations in sizes. These images maintain character sizes within a range of approximately 8-10 units.

A range of sample images extracted from our thoughtfully curated Telugu language dataset is displayed in Table IV. This dataset was created from a diverse range of sources, encompassing inputs from friends, family, and members of the GRIET community.

The pie chart in Fig. 6 offers a clear picture of how word lengths in our curated Telugu dataset are distributed in percentages. It's fascinating to note that a significant 67.1

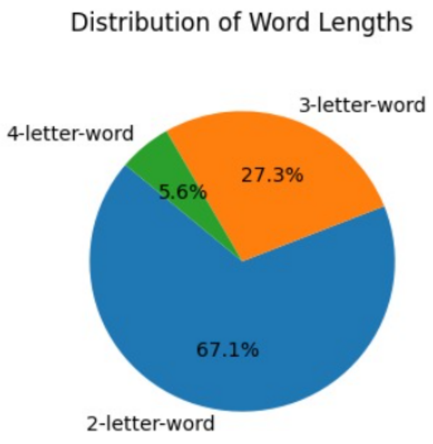


Fig. 6. Pie Chart

percentage of the words are composed of only two letters, followed by 27.3 percentage for three-letter words, and a smaller 5.6 percentage for four-letter words. What makes this observation intriguing is that, due to the abundance of two-letter words, the model is particularly adept at interpreting images containing these short word structures.

The bar chart in Fig. 7 paints a clear picture of how word lengths in our curated Telugu dataset are distributed in terms of frequency.

What's particularly interesting is that a significant number, 2,013 words to be exact, are made up of just two letters. Following closely behind are three-letter words with 819 occurrences, and four-letter words, which are less common at 168 instances.

The bar chart in Fig. 8 offers a straightforward view of the distribution of word length classes of our curated Telugu dataset by their frequency.

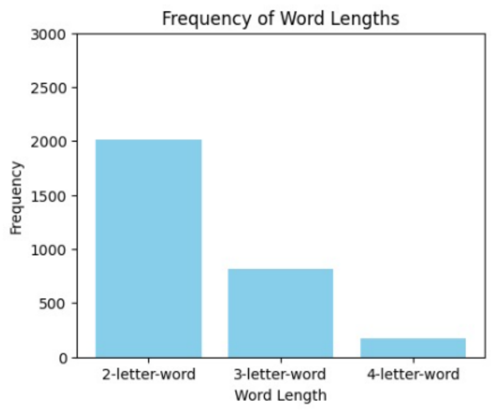


Fig. 7. Bar Graph

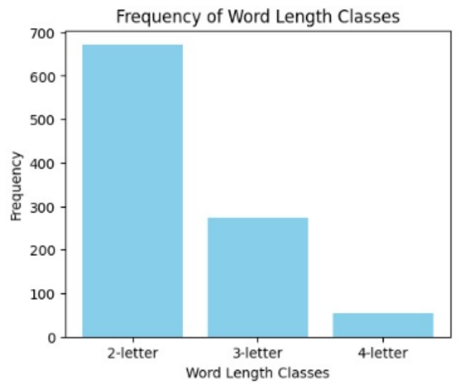


Fig. 8. Bar Graph

What stands out as particularly intriguing is the substantial presence of 671 classes consisting of just two letters. Close behind are three-letter classes with 273 occurrences, while four-letter classes are notably less frequent, with only 35 instances. It's quite evident that the substantial presence of two-letter words naturally results in a significant number of corresponding two-letter word classes.

V. CONCLUSION AND FUTURE WORKS

This paper contributes significantly to the fields of computer vision and natural language understanding by introducing a novel approach that integrates synthetic data generated by pre-trained language models with predicted global semantic information. This integration enhances the decoder's ability to handle challenging scenarios effectively, particularly when faced with incomplete or blurry word images, thus ensuring accurate predictions and overcoming ambiguity. Additionally, our methodology addresses the issue of image noise, providing a reliable guide for the decoding process. These advancements hold promise for improving the accuracy and robustness of text recognition systems, offering significant benefits to various applications and industries.

TABLE III
SAMPLE IMAGES IN IIIT-INDICHW-WORDS[22] DATASET

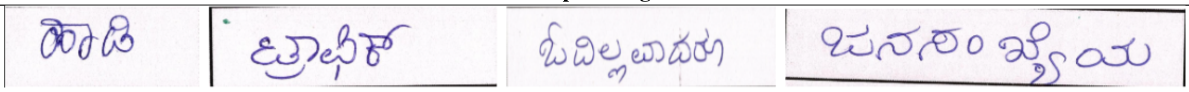
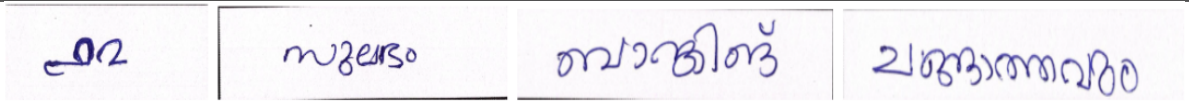
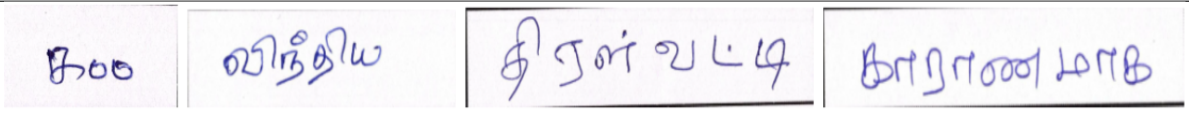
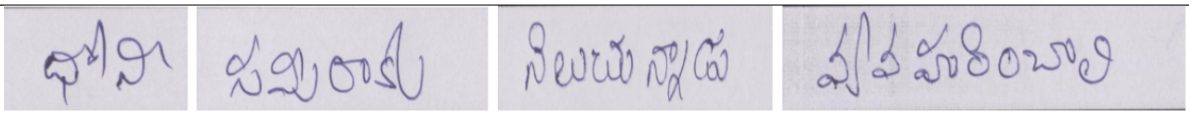
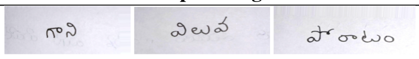
Languages	Sample Images
Kannada	
Malayalam	
Tamil	
Telugu	

TABLE IV
SAMPLE IMAGES FROM OUR DATASET

Languages	Sample Images
Telugu	

Continued research and development will unleash the full potential of this approach. It offers revolutionary possibilities for automated text recognition, language translation, sentiment analysis, voice assistants, and more. The versatility of this method allows adaptation to specific languages and dialects, making it indispensable for linguistic diversity.

ACKNOWLEDGMENT

We thank the Department of Science and Technology(DST), Government of India for providing financial support for our project (Sanction Number: SP/YO/2021/2096 (C & G)).

REFERENCES

[1] Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). ImageNet Classification with Deep Convolutional Neural Networks.

[2] Kingma, D. P., & Welling, M. (2013). Auto-Encoding Variational Bayes. arXiv preprint arXiv:1312.6114.

[3] Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., & Bengio, Y. (2014). Generative adversarial nets. In Advances in neural information processing systems.

[4] Karras, T., Aila, T., Laine, S., & Lehtinen, J. (2018). Progressive Growing of GANs for Improved Quality, Stability, and Variation.

[5] Turk, M., & Pentland, A. (1991). Eigenfaces for recognition. Journal of Cognitive Neuroscience, 3(1), 71-86.

[6] Xie, Q., Dai, Z., Hovy, E., Luong, M. T., & Le, Q. V. (2019). Unsupervised Data Augmentation.

[7] Zhang, H., Cisse, M., Dauphin, Y. N., & Lopez-Paz, D. (2017). MixUp: Beyond Empirical Risk Minimization.

[8] Krishnan, P., & Jawahar, C. V. (2016). Generating Synthetic Data for Text Recognition. arXiv e-prints. doi:10.48550/arXiv.1608.04224.

[9] Roy, Partha, Mohta, Akash, & Chaudhuri, Bidyut. (2018). Synthetic data generation for Indic handwritten text recognition.

[10] Le, Ha, Kim, S. H., Na, In, Do, Yen, Park, Sang-Cheol, & Jeong, Sun-Hwa. (2012). Automatic Generation of Training Character Samples for OCR Systems. International Journal of Contents, 8, 83-93. doi:10.5392/IJoC.2012.8.3.083.

[11] O'Shea, Keiron Nash, Ryan. (2015). An Introduction to Convolutional Neural Networks. ArXiv e-prints.

[12] Saini, Naresh, Pinto, Promodh, Bheemaraj, Aravinth, Kumar, Deepak, Daga, Dhiraj, Yadav, Saurabh, & Nagaraj, Srihari. (2022). OCR Synthetic Benchmark Dataset for Indic Languages.

[13] Manousakas, Dionysis, & Aydore, Sergul. (2023). On the Usefulness of Synthetic Tabular Data Generation.

[14] Kang, Lei, Rusiñol, Marçal, Fornés, Alicia, Riba, Pau, & Villegas, Mauricio. (2019). Unsupervised Writer Adaptation for Synthetic-to-Real Handwritten Word Recognition.

[15] Varga, Tamás, & Horst Bunke. (2003). Effects of training set expansion in handwriting recognition using synthetic data. Proc. 11th Conf. of the Int. Graphonomics Society.

[16] Dutta, K., Krishnan, P., Mathew, M., & Jawahar, C. (2018). Offline handwriting recognition on Devanagari using a new benchmark dataset. In DAS (pp. 25–30).

[17] TAŞDEMİR, Esma Fatıma BİLGİN. (2021). Online Turkish handwriting recognition using synthetic data. Avrupa Bilim ve Teknoloji Dergisi, 32, 649-656.

[18] Johannes Michael, Roger Labahn, Tobias Grüning, & Jochen Zöllner. (2019). Evaluating sequence-to-sequence models for handwritten text recognition. In ICDAR (pp. 1286–1293).

[19] Bojanowski, P., Grave, E., Joulin, A., & Mikolov, T. (2017). Enriching word vectors with subword information. Transactions of the Association for Computational Linguistics, 135–146.

[20] Kumar, K. Vijay, & R. Rajeshwara Rao. (2013). Online handwritten character recognition for Telugu language using support vector machines. International Journal of Engineering and Advanced Technology, 3(2), 189-192.

[21] Jaderberg, Max, Karen Simonyan, & Andrew Zisserman. (2015). Spatial transformer networks. Advances in neural information processing systems 28.

[22] Gongidi, S., Jawahar, C. (2021). IIIT-INDIC-HW-WORDs: A dataset for Indic handwritten text recognition. In ICDAR (pp. 444–459).