# PKIT: Printed Kashmiri Image Text Recognition Using Deep Learning

[1]Maajid Bashir,[2]Vishal Goyal, [3]Kaiser J.Giri

[1]Department of Computer Science, Punjabi University, Patiala,147002, Punjab, India.

[2]Department of Computer Science, Punjabi University, Patiala, 147002, Punjab, India.

[3]Department of Computer Science, Islamic University of Science and Technology, Awantipora, 192122, Jammu and Kashmir, India

[1]maajid.net@gmail.com, [2]vishal.pup@gmail.com, [3]kaiserjaveed@gmail.com;

## ABSTRACT

Optical Character Recognition, often known as OCR, is a method that turns scanned documents, images of text, and PDFs into text documents, that can be edited and searched on a computer. OCR software analyzes a scanned image of text and turns it into machine-encoded text by identifying the characters in the image and transforming them into a digital format. Acknowledging the relevance of optical character recognition (OCR) in the actual world, a multitude of approaches have evolved both for Western and Asian languages. Kashmiri is mostly spoken in the Kashmir Valley, which is located in Jammu and Kashmir India. In spite of the significant amount of effort that has been done into recognizing Indian scripts such as Devanagari, Bengali, Urdu, and Punjabi, no such effort has been made to recognize Kashmiri script. In addition, several benchmark corpora for other Perso Arabic scripts, such as Urdu, Arabic, and Pashto, have been developed for the purpose of training and assessing various OCR systems. Notably, there is currently no OCR corpus for Kashmiri script that can be utilized to train and evaluate deep neural networks for the development of Kashmir OCR. To that purpose, we have proposed a Kashmiri corpus Printed Kashmiri Image Text (PKIT)consisting of 120000 line, and 523000-word level printed text images respectively, well suited for use in deep learning techniques. Additionally, we used the proposed dataset for training different state of art deep learning approaches thereby obtaining the Word Error Rate (WER) and Character Error Rate ((CER)of 5.62% on average.

Keywords: Dataset generation, Deep learning, Kashmiri OCR, optical character recognition, Printed Kashmiri text recognition

## INTRODUCTION

Optical Character Recognition (OCR) is a method for extracting text from images of printed or handwritten text [1, 2]. OCR systems are commonly accepted to have a wide range of applications. An OCR, for example, can read and convert text from a scanned image of a newspaper or book to a set of Unicode characters, which can be used for a variety of tasks like highlighting, searching, annotating, and translating [3]. Moreover, OCR's application
domains encompass both government and private companies. For example, it can be used by the City Traffic Police to recognize license plates [4, 5], the Immigration Department to recognize passports, and banking institutions to process demand drafts and cheques automatically [6]. Besides, OCR may be utilized to preserve and digitize historical texts. OCR is generally recognized as an established research challenge in the area, and as a result, a significant number

of rules-based and supervised learning algorithms have been proposed to address it [3].
OCRs fall within a variety of categories. For example, optical character recognition (OCR) can be either online or offline [7]. Whereas, Online OCRs, on the other hand, can recognize text in real-time by analyzing patterns made with a pen or painted on a touch screen panel [3, 8]. Offline OCRs, on the other hand, work with text that has already been retrieved from an image [3, 8]. The development of offline OCRs is considered to be a more challenging problem than online ones.
Additionally, Segmentation-based OCRs and segmentation-free OCRs are the two main categories that may be used to classify OCR techniques. The first method requires the text to be segmented for training and prediction at either the ligature or character level, whereas the second method does not need any segmentation of the text at all [9]. Generally, segmentation-based devices require a character image that has been cropped using some segmentation approach [10, 11]. Without previous segmentation, a segmentation-free OCR may recognize full phrases or text lines. Moreover, segmentation-based OCRs require fewer data than segmentation-free OCRs for better results. Furthermore, OCRs based on segmentation need individual character tagging inside images, which is a harder job. OCR technology has evolved in the last several years due to the advances in deep learning technologies. Several research has employed deep learning approaches to develop OCRs for the Latin script, such as English, French, and German, with extraordinarily high accuracies [12, 13]. Asian languages, on the other hand, differ substantially from European languages both in morphology and direction of writing. English, French, and German are, for example, written left to right, but Arabic-adapted scripts like Kashmir, Urdu, and Persian are written from right to left. Which adds to the complexity of recognition of text from Arabic-adapted scripted languages. Numerous efforts have been made to develop language-specific optical character recognition (OCR) systems for Arabic, Urdu, Persian, Hindi, and Punjabi [14, 15]. However, no effort on developing an end-to-end Kashmiri optical recognition system has been undertaken. To that purpose, the following major contributions are made in this study:

1. To begin, we developed a comprehensive Kashmiri optical character recognition (OCR) dataset known as PKIT. This dataset is enough to support the development of a neural network model. The stated dataset has a total of 6,43,000 image files, each of which is comprised of 120000 text line images and 523000 words.
2. In the second phase of the study, two different state-of-art deep learning models are applied for the elevation of the proposed corpus.

The rest of the paper is organized as follows: Section 2 outlines various Deep Learning techniques. The Peculiarities of the Kashmiri script are discussed in Section 3. The related work is summarized in section 4. In Section 5, the procedure for creating the corpus is presented. The approach that has been suggested has been covered in section 6. The results of the experiments and their discussion are outlined in section 7. In the last section of this paper, section 8, a conclusion is provided along with the identification of numerous significant areas for the direction of future study.

## DEEP LEARNING

Deep Learning is a branch of machine learning that employs multi-layer neural networks to analyse and comprehend complex information such as images, audio, and text. The networks are designed to automatically learn from the input data, leading to high accuracy for tasks such as voice and image recognition. Rather than being programmed explicitly, the neural networks aim to learn characteristics from the data as it is fed into them, allowing them to perform complex tasks with a high degree of precision. Deep learning has achieved remarkable results in various domains like computer vision, natural language processing, and autonomous vehicles. This approach can be used in supervised, unsupervised, or semi-supervised settings. Supervised Deep Learning techniques involve training a neural network using labelled data where the desired output (label) is provided for each input. Supervised deep learning methods are more advanced than unsupervised methods. The network is trained to map the inputs to the outputs based on the examples that have been labelled, and it may then be used to generate predictions based on inputs that it has not previously seen. Some examples of supervised deep learning include deep neural networks (DNNs), convolutional neural networks (CNN) and recurrent neural networks (RNN) like LSTM and GRU. Supervised deep learning algorithms include the use of deep neural networks (DNN), convolutional neural networks (CNN), and recurrent neural networks (RNN) such as long short-term memory (LSTM) and Gated Recurrent Units (GRU). In contrast, unsupervised deep learning involves training a neural network with no labelled output. However, the network may be utilized for dimensionality reduction, clustering, and anomaly detection after being trained to understand the underlying structure or distribution of the data. Unsupervised deep learning techniques clustering, dimensionality reduction, and generative methods such as Autoencoders (AEs), Restricted Boltzmann Machines (RBMs), and Generative Adversarial Networks (GANs). Semi-supervised learning is a technique that combines elements of both unsupervised and supervised learning and aims to improve the model performance using a limited set of labelled data and a larger amount of unlabelled data. Reinforcement Learning is a form of deep learning where an agent learns to take a sequence of actions by optimizing a reward signal through interactions with its environment.

## CONVOLUTION NEURAL NETWORKS (CNN)

CNNs are multi-layered networks that are trained to detect patterns in data and are optimized for working with data having a grid-like structure, such as an image. networks are programmed to automatically and dynamically learn the hierarchical spatial features of the data. They are very helpful for image recognition because they are able to analyze the data using numerous layers of calculations, each of which derives numerous features from the image. This makes them extremely valuable for image recognition. Furthermore, CNNs are widely utilized in various applications, including image classification, video analysis, and object recognition. A Convolutional Layer is the basic building block of a CNN. The convolution operation, which is defined by the convolution equation, is the key calculation performed in a layer dedicated to convolution processing. It involves two functions, the input image and the kernel (also known as a filter), and produces a third function, the output feature map. This mathematical process starts by sliding the kernel across the input image and performing element-wise multiplication between the kernel and the overlapping portion of the image, followed by summing the results. The calculation of subsequent feature maps is given by the equation W and A representing the input image and kernel, respectively, (p,q) denoting the current position of the kernel on the image, and (r,s) indicating the position of a pixel within the input image.

$$(W * A)(p, q) = \sum_{r} \sum_{s} W(r, s)\, g\,(p - r, q - s) \qquad (1)$$

Following the convolution operation, the output feature map is then processed through a non linear activation function, such as ReLU, element-wise.

$$O(p, q, k) = \max(0, O(r, s, k)) \tag{2}$$

where O (r, s, k) is the kth output filters feature map at location (p, q) The output feature map from the activation function is then subjected to a pooling operation, such as max pooling, which reduces the spatial dimension of the feature map while retaining the most significant information.

$$O'(p, q, k) = \max(O(is : is + r, js : js + r, k)) \tag{3}$$

The stride s is the step size of the pooling operation, and p represents the size of the pooling window. The primary objective of pooling is to reduce the computational load within the network by reducing the total number of parameters and the risk of overfitting. The result from the pooling operation is then fed into one or multiple fully connected layers, which operate similarly to the hidden layers in a standard neural network. In the fully connected layers, the input is transformed into the output through a linear operation represented by the equation

$$y = W * X + b \tag{4}$$

where Y is the output, X is the input, W is the matrix of weights for the fully connected layer, and b represents the bias term. These layers analyze the features generated by the previous layers to make predictions or classify inputs. The general representation of a CNN is depicted in Fig. 1

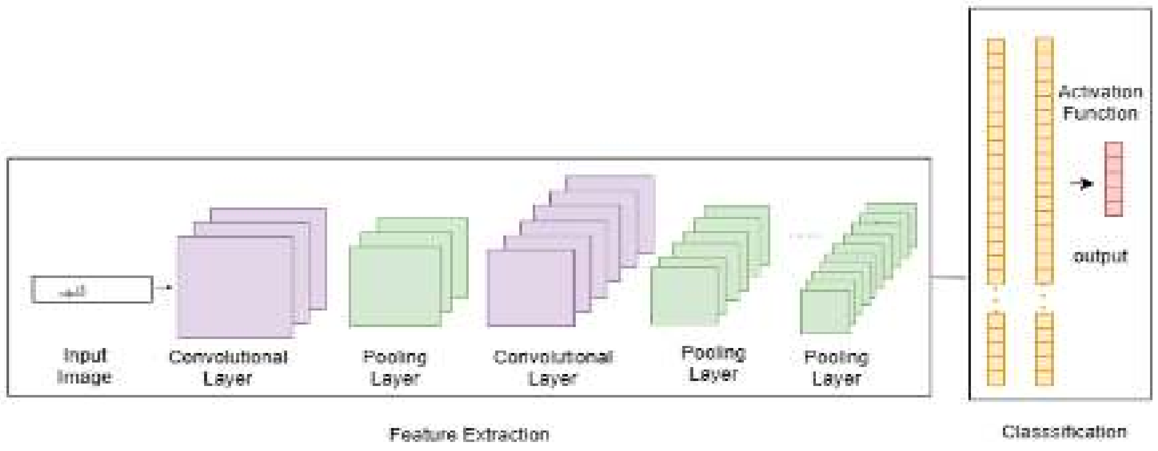


Fig. 1 General Architecture of the CNN

## RECURRENT NEURAL NETWORKS (RNN)

Recurrent Neural Networks (RNNs) are specifically built to process sequential data such as text, audio, or time series. Unlike standard neural networks that treat each input as separate, RNNs have cyclic connections that enable information to be transferred throughout the network over time.

An RNN may be viewed as a network of memory cells that are interconnected throughout time. After receiving the most recent input and the current state of the network as input, each memory cell produces an output and a new state for the network. The newly learned state is then sent to the subsequent memory cell in the sequence, and so forth.
The behavior of an RNN at a specific time step t is defined by mathematical equations that relate its input (represented by x(t)), hidden state (represented by h(t)), and output (represented by y(t)). The input, hidden state, and output each have corresponding bias vectors (bx, bh, by) and weight matrices (Wx, Wh, Wy), and the hidden state at time t is determined by combining the input at time t and the previous hidden state.

$$h(t) = \sigma(Wx*x(t) + Wh*h(t-1) + bh) \tag{5}$$

The computation of the output at time step t involves the use of the hidden state at the same time step t and is expressed as:

$$y(t) = Wy * ht + by \tag{6}$$

Afterwards, the SoftMax function is applied to the outputs to produce a probability distribution. There are different variations of RNN architecture, such as LSTM and GRU, each of which has its specific set of mathematical equations. The overall structure of RNN is illustrated in Fig. 2

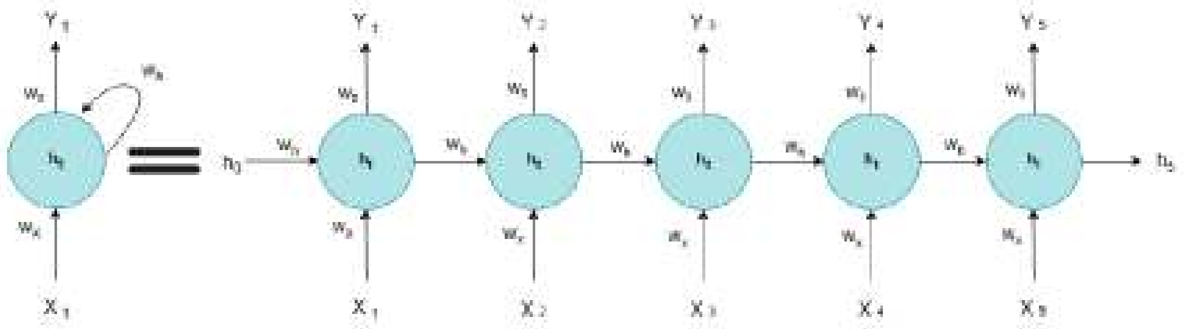
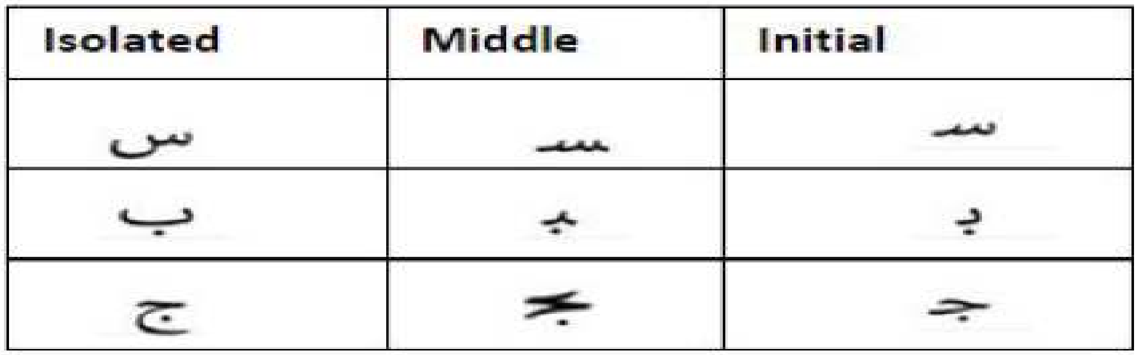
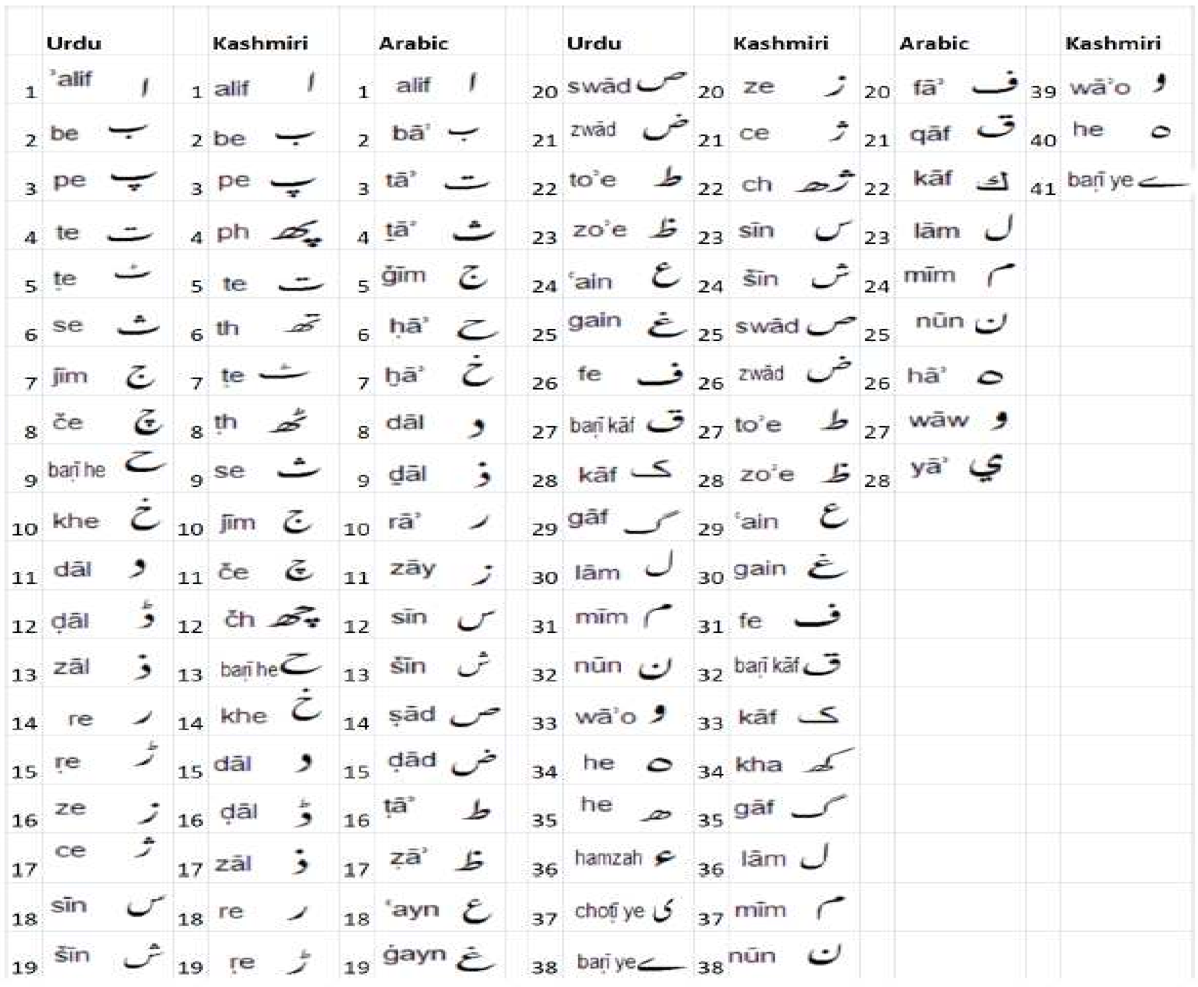


Fig. 2 Architecture of the RNN



| Isolated | Middle | Initial |
|---|---|---|
| س | ـسـ | سـ |
| ب | ـبـ | بـ |
| ج | ـجـ | جـ |

Fig. 3 Shape of the Kashmiri alphabet based on position within the text.

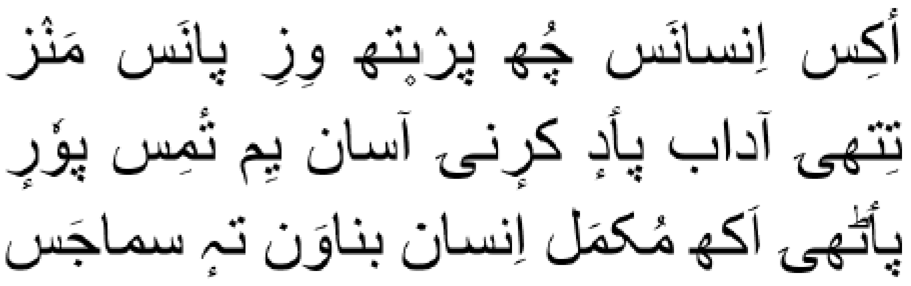


| | Urdu | | Kashmiri | | Arabic | | Urdu | | Kashmiri | | Arabic | | Kashmiri |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | ʾalif ا | 1 | alif ا | 1 | alif ا | 20 | swād ص | 20 | ze ز | 20 | fāʾ ف | 39 | wāʾo و |
| 2 | be ب | 2 | be ب | 2 | bāʾ ب | 21 | zwād ض | 21 | ce ژ | 21 | qāf ق | 40 | he ہ |
| 3 | pe پ | 3 | pe پ | 3 | tāʾ ت | 22 | toʾe ط | 22 | ch ژھ | 22 | kāf ك | 41 | baṛī ye ے |
| 4 | te ت | 4 | ph پھ | 4 | ṯāʾ ث | 23 | zoʾe ظ | 23 | sīn س | 23 | lām ل | | |
| 5 | ṭe ٹ | 5 | te ت | 5 | ǧīm ج | 24 | ʿain ع | 24 | šīn ش | 24 | mīm م | | |
| 6 | se ث | 6 | th تھ | 6 | ḥāʾ ح | 25 | gain غ | 25 | swād ص | 25 | nūn ن | | |
| 7 | jīm ج | 7 | ṭe ٹ | 7 | ḫāʾ خ | 26 | fe ف | 26 | zwād ض | 26 | hāʾ ه | | |
| 8 | če چ | 8 | ṭh ٹھ | 8 | dāl د | 27 | baṛī kāf ق | 27 | toʾe ط | 27 | wāw و | | |
| 9 | baṛī he ح | 9 | se ث | 9 | ḏāl ذ | 28 | kāf ک | 28 | zoʾe ظ | 28 | yāʾ ي | | |
| 10 | khe خ | 10 | jīm ج | 10 | rāʾ ر | 29 | gāf گ | 29 | ʿain ع | | | | |
| 11 | dāl د | 11 | če چ | 11 | zāy ز | 30 | lām ل | 30 | gain غ | | | | |
| 12 | ḍāl ڈ | 12 | čh چھ | 12 | sīn س | 31 | mīm م | 31 | fe ف | | | | |
| 13 | zāl ذ | 13 | baṛī he ح | 13 | šīn ش | 32 | nūn ن | 32 | baṛī kāf ق | | | | |
| 14 | re ر | 14 | khe خ | 14 | ṣād ص | 33 | wāʾo و | 33 | kāf ک | | | | |
| 15 | ṛe ڑ | 15 | dāl د | 15 | ḍād ض | 34 | he ہ | 34 | kha کھ | | | | |
| 16 | ze ز | 16 | ḍāl ڈ | 16 | ṭāʾ ط | 35 | he ھ | 35 | gāf گ | | | | |
| 17 | ce ژ | 17 | zāl ذ | 17 | ẓāʾ ظ | 36 | hamzah ء | 36 | lām ل | | | | |
| 18 | sīn س | 18 | re ر | 18 | ʿayn ع | 37 | choṭī ye ی | 37 | mīm م | | | | |
| 19 | šīn ش | 19 | ṛe ڑ | 19 | ġayn غ | 38 | baṛī ye ے | 38 | nūn ن | | | | |

Fig. 4 Alphabet set for Urdu, Kashmiri and Arabic

## PECULARITIES OF KASHMIRI LANGUAGE

Kashmiri is spoken by about 7 million people, mostly in the Indian administered territory of Jammu and Kashmir, as well as a few hundred



أکِس اِنسانَس چھُ پرٛیتھ وزِ پانَس مَنٛز
تِتھی آداب پأدِ کرِنی آسان یِم تٔمِس پوٗرِ
پأٹھی اَکھ مُکمَل اِنسان بناوَن تہٕ سماجَس

Fig. 5 Example of Kashmiri script

thousand in Pakistan and other regions. Kashmiri is an Indo-Aryan language belonging to the Dardic subgroup of the Indo-Aryan language family. Under Schedule VIII of the Indian Constitution, the Kashmiri language has been selected as one of the 22 Indian languages whose development the government will focus on. To promote the Kashmiri language, the Jammu and Kashmir government has made it a compulsory subject in primary schools up to class VIII[16].
The Kashmiri script is a superset of Arabic and has a total of 41 characters, 13 of which are additional to the Arabic alphabet. Fig. 4 depicts the alphabet set that is used in the writing systems of Urdu, Kashmiri, and Arabic. Kashmiri includes a multitude of consonants and vowels than Arabic as well as Urdu. Kashmiri script employs a bidirectional writing system similar
to Arabic and Urdu script, which is written and read in the right-to-left direction. The letters are written from right to left, whereas the numerals are written from left to right [17]. Kashmiri inherits the complexity of the Urdu and Arabic writing systems. In addition to that, the linguistic characteristics of Kashmiri, such as the presence of central vowels that are not found in any other Indo-Aryan language and a richer character set, contribute to the complexity of the language [16]. In addition to that, the linguistic characteristics of Kashmiri, such as the presence of central vowels that are not found in any other Indo-Aryan language and a richer character set, contribute to the complexity of the language.
The cursive nature of the Kashmiri writing system poses the main challenge in the recognition task. The alphabets are written in cursive starting at the right and moving to the left. In this regard, alphabets in the Kashmiri script are joined with a word or adjacent characters. However, a word's characters may or may not be connected together to create ligature. Moreover, Characters and ligatures may overlap in the Kashmiri script, making it a challenge for segmentation. Fig. 5 is an example of the Kashmiri script.
The existence of filled loops in several characters presents another challenge in the the recognition of the Kashmiri script. This makes the process of character identification a more difficult task. Because, while performing recognition, it becomes quite difficult to discern between the several characters that have the same shape, such as wao and dal.
The context-sensitive nature of the Kashmiri script adds to the complexity of recognition characters, as characters change their position based on the location (initial, middle, or isolated) within the word as depicted in Fig. 3. An in-depth study of these issues may be found in [16].

## RELATED WORK

Numerous research has been undertaken to create optical character recognition (OCR) systems for Western and Asian languages. Moreover, multiple benchmark datasets for these languages are now accessible. On the contrary, no efforts have been made to build Kashmiri OCR. This section provides an overview of studies that have been conducted to recognize Arabic adapted scripts such as Arabic and Urdu.
When developing OCRs for Arabic-adapted scripts like Urdu and Arabic, deep learning-based methods have historically outperformed traditional machine learning approaches (SVMs). The first study using deep learning methods for Arabic text recognition dates back to [18]. In that study, Arabic text was identified in images using the MDLSTM network along with the CTC loss function. The accuracy of the model, which was reported to be 91.4%, In the past, researchers used many types of deep learning algorithms to work with Urdu datasets. For example, they've used it to work with character and word classifiers, feature extraction tools, and even as an end-to-end OCR application for the Urdu language has been developed. tacked

Denoising Autoencoders, for example, that were initially employed to recognize Bangla [19] were also utilized to recognize ligatures of Urdu script [20].
In the study, [21] an end-to-end Urdu OCR was built using a hybrid approach by combining CNN with RNN. In addition to that, OCRs that were suggested were based on Layers of LSTM and the CTC loss function.
In the study [22] proposed a neural network-based online handwriting system that could recognize Arabic letters. The proposed system is largely focused on preprocessing, feature extraction, and classification. In the first step, all characters are preprocessed to enhance the product's overall visual quality. Next, the image of each character is transformed into a 2-bit image (binary image). Afterward, during the training of the neural network, the backpropagation technique was used. The proposed study claims an accuracy of 83%.
In the work, [23] a scanned image of the Arabic Naksh script is subjected to preprocessing, word-level feature Extraction, character segmentation, recognition, and post processing. This research also presents methods for word and line segmentation. Finally, the study proposes a neural network character-based recognition model evaluated on open-source Arabic corpus datasets such as WATAN and APTI. The authors have claimed accuracy of 98.66% and 99.89% on character segmentation.
A similar study [24], proposes a model that can read Arabic text printed in a variety of typefaces, including some that are designed to seem like handwritten Arabic. The suggested approach makes use of a hybrid DL network to automatically detect written Arabic text. The algorithm was put to the test on a customized dataset consisting of over 2 million-word samples written in 18 distinct Arabic fonts. The goal of the evaluation was to see how well the model could identify a variety of Arabic typefaces reflecting different handwriting styles.
In the paper [25], the authors have introduced a novel dataset Hijja, which is comprised of Arabic letters written only by youngsters between the ages of 7 and 12. The proposed dataset includes 47,434 different characters contributed by 591 different individuals. In addition, they presented a CNN-based model for the automated identification of handwritten Arabic characters. The proposed model has been trained on Hijja and the Arabic Handwritten Character Dataset (AHCD) dataset. The authors claim that their methodology has an accuracy of 88% on the Hijja dataset and 97 % on the AHCD dataset respectively.
In another similar type of study [26], a handwritten Urdu character dataset for the Nasta'liq script includes both isolated and positioned characters, as well as numbers. The paper also suggests a CNN architecture for identifying handwritten Urdu letters and numerals, with an accuracy of 98.82%.
The authors of this study [27] propose an end-to-end Arabic text recognition system. It uses vision transformers as encoders (BEIT) and vanilla transformers as decoders, and it replaces the use of convolutional neural network (CNN) architecture for feature extraction. As a result, the complexity of the model is simplified. According to the author's claims, the character error rate (CER) was 4.2%.
Based on a review of the related work, it was discovered that no effort has been taken to develop Kashmiri OCR. Our research provides some first steps in this field of study and gives toolkits that may be used to further develop and expand the scope of the project.

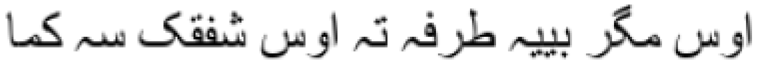
## DATASET GENERATION

This section presents how our novel Kashmiri OCR dataset (PKIT) is built. There are two levels in our proposed dataset, from the line level down to the word level. A sample image of the text line and words from the proposed dataset (PKIT) are depicted in Fig. 6 and Fig. 7 respectively. In the process of creating an OCR corpus for Arabic-adapted scripts such as Arabic, Persian, and Urdu, three distinct methods have been used: controlled, original and synthetic. The controlled procedure is used for developing a handwritten OCR data set. The individuals are instructed to replicate a digitally available text in their handwriting. Following that, the textual documents are scanned to create an image corpus, with the original text in digital to use for the labeling of these images. This method's core guarantee is that the human standard dataset may be developed with relatively little work required. In addition, by carefully selecting topics, this method allows for manageable variation in writing. The fact that it is exceedingly challenging to build a dataset that is sufficiently big and acceptable for cutting-edge deep learning algorithms is the most important disadvantage of using this approach. This is a result of the fact that it is a task that necessitates a considerable amount of time to organize a sufficient number of subjects and generate adequate samples from each resource.

The real-world method begins with the collection of the already-existing documents, followed by the generation of images by scanning documents, and last, the production of the ground truth, also known as the human benchmark [27]. In existence, texts might be in the form of either handwritten notes or printed books and newspapers. One of the method's key benefits seems to be how closely the constructed corpus mimics real-world cases. However, the approach is not without its share of downsides. Making a digital human reference using images and the corresponding Urdu text is a laborious process. Second, there just aren't enough resources covering all the areas.

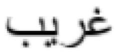
The synthetic method involves inputting unprocessed digital text into a computerized system so that it can produce images of printed text. The rapid production of a massive corpus along with the ground truth that goes along with it is a significant advantage of using this technique. Although data augmentation techniques such as rotation, contrast, flipping, scaling, cropping, and other degradations can be added computationally, one of the potential drawbacks of using this approach is that the resulting images could not have enough of the essential diversity that is often present in setups that are derived from the actual world. The proposed dataset PKIT is generated using a synthetic technique owing to its capacity to swiftly build a large image corpus along with the associated label. Furthermore, the proposed dataset includes no degraded versions of images. Thus, augmenting the dataset may expand its size and variety as desired.

The procedure used to create the PKIT corpus is depicted in Fig. 8. The entire procedure is divided into three stages: data cleaning, tokenization, and the generation of fixed-size text images along with the text file containing the labels of corresponding images.



اوس مگر بییہ طرفہ تہ اوس شفقک سہ کما

Fig. 6 Text line image



غریب

Fig. 7 Word level image

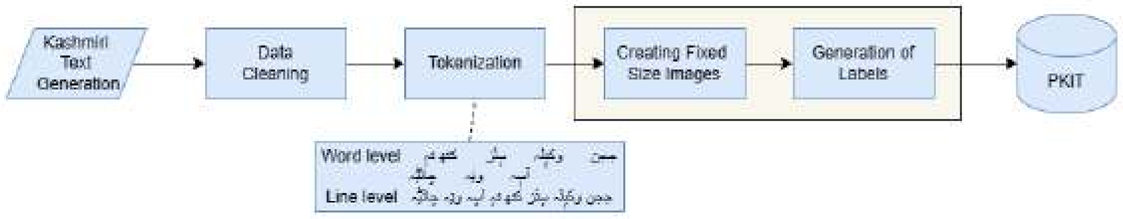


Fig. 8 Dataset generation process

## DATA CLEANING

As depicted in Fig. 8. The raw Kashmiri text file serves as input. This raw text was obtained from the Central Institute of Indian Languages (CIIL) having a Corpus size of 466,054 Words and a character count of 26,46948 from various sources like newspapers, and magazines.
The acquired text needed to be preprocessed further after having non-Kashmir content removed. To achieve this end, a two-step process for data cleansing was implemented. Before anything else, the HTML tags were stripped out using the in-built regular expression-based text-matching functionality in Python namely RegEX. Additionally, the Unicodes used to create emoticons were removed from the text using a parsing process. The second phase included creating an index of Unicodes that included the frequency with which each character occurred in the unprocessed data. Using this the Unicodes that appeared less often are treated as invalid values and were removed to obtain the cleaned Kashmiri raw text.

## TOKENIZATION

The second step in creating a corpus is to produce Kashmiri text fragments that may be utilized later on to create an image corpus. As was said before, our goal is to create a word- and line-level image corpus. To do this we need need to apply 2- levels of tokenization. The specifics of each process are presented below:
Too proceed with the tokenization process, the first step requires the selection of Kashmiri text fragments that have been separated into lines of text. A text line is a logical series of words conveying information, it may or may not be a whole sentence. Text lines may be broken up into many sentences. In this stage, we used the symbol hash # as a delimiter for the extraction of text lines from the obtained Kashmiri text. After that, many post-processing procedures were executed, such as separating lines of text that were either blank or too short (consisting of three words or fewer) or had too few characters in total (less than eight). As a result, 260000 potential text lines were selected for use in the second step, for the creation of text line images.
For creating word level, the corpus was tokenized at the word level by employing white spaces as delimiters for separating individual words. A total of 6230000 words, were found in the cleaned text for use in creating images of Kashmiri words.

## IMAGE GENERATOR

In this step, we create fixed-size images for use in the development of Kashmiri OCR for printed text. More specifically, we want to compile a dataset of images at the line and word levels.

Additional analysis of the 260000 potential textline extracts was undertaken to generate the text-line image corpus. Each line of text was measured and averaged, as well as compared to the shortest and longest possible line. Furthermore, the length distribution and standard deviation (SD) were calculated. According to the findings, it was found that the text line with the fewest words consisted of just four, while the line with the most words included twenty. The existence of such a wide range of variances highlights the necessity of meticulously selecting text lines and precise image dimensions. The purpose of employing a fixed-size image generator instead of a variable-size image generator was to produce an OCR corpus that could be immediately utilized without undergoing any prior processing before inputting it into a deep neural network. The proposed dataset consists of line images with a resolution of 128 by 40 pixels and word images with a resolution of 32 by 32 pixels and font size of 12 points. Each image has a white background, black foreground, and centered text fragments. Consequently, 6,43,000 images with 1,20,000 text lines and 523,000-word images were created. Sample image from the proposed dataset for both line and word level are depicted in Fig .6 and Fig. 7 respectively.

## GENERATION OF LABELS

During this phase, labels were produced for each of the 6,43,000 images in the dataset. This was done in order to offer a format that is intelligible by both humans and machines, which may help future research that makes use of the PKIT. The label consists of a name and associated text within the image.

## DEEP LEARNING APPLIED ON PKIT DATASET

To assess the proposed PKIT, we employ deep learning-based techniques. The goal of this research is to evaluate the recognition accuracy of the proposed corpus at both line and word levels. In subsequent sections, we offer a thorough explanation of each proposed method.

Our approach employs convolutional neural networks (CNN) along with recurrent neural network (RNN), and connectionist temporal classification (CTC) loss to recognize text from images, as recommended in [28]. CNN has been used to extract the features from Kashmiri script images to be given as input to GRU, which in turn generates a series of probabilities at every time stamp to be given as a parameter to the CTC loss function for predicted.

## CONVOLUTION NEURAL NETWORK (CNN) + GATED RECURRENT UNIT ARCHITECTURE

Fig. 9 shows a proposed architecture of CNN+GRU, which has been impacted by the existing model [28].

Both word recognition and text line image recognition were trained on separate but equivalent iterations of the same model. In the initial version, the model was trained to infer text from a single-word visual. In the second variant, text was predicted based on the line image, which included more than one word. The specifications of the model are outlined in Table 1.

## VGG-16 + GATED RECURRENT UNIT (GRU)

The VGG1-6 model is a type of CNN that was trained using data from the ImageNet dataset. It was developed by a group at the University of Oxford named the Visual Geometry Group. It is a common practice to utilize this model or a derivative of it when tasked with classifying images. The paper [29] made the first presentation of it. The model is comprised of 16 convolutional layers and is distinguished by its straightforward design as well as its strong performance. Convolutional Neural Networks (CNNs) are commonly used as a benchmark model for comparison with more advanced techniques. Fig. 10 shows a proposed architecture of CNN+BLSTM. The model's specs are shown in Table 2.

| CNN | | | | | | | |
|---|---|---|---|---|---|---|---|
| Layer | #Filters | Kernel size | Stride | Padding | Activation | eps | momentum |
| Conv2d | 64 | (3,3) | (1,1) | (1,1) | LeakyReLU | — | — |
| BatchNorm2d | 64 | — | — | — | — | 1e − 05 | 0.1 |
| MaxPool | — | (2,2) | (2,2) | 0 | — | — | — |
| Conv2d | 128 | (3,3) | (1,1) | (1,1) | LeakyReLU | — | — |
| BatchNorm2d | 128 | — | — | — | — | 1e − 05 | 0.1 |
| MaxPool | — | (2,2) | (2,2) | 0 | — | — | — |
| Conv2d | 256 | (3,3) | (1,1) | (1,1) | LeakyReLU | — | — |
| BatchNorm2d | 256 | — | — | — | — | 1e − 05 | 0.1 |
| Conv2d | 256 | (3,3) | (1,1) | (1,1) | LeakyReLU | — | — |
| BatchNorm2d | 256 | — | — | — | — | 1e − 05 | 0.1 |
| MaxPool | — | (2,1) | (2,1) | 0 | — | — | — |
| Conv2d | 512 | (3,3) | (1,1) | (1,1) | LeakyReLU | — | — |
| BatchNorm2d | 512 | — | — | — | — | 1e − 05 | 0.1 |
| Conv2d | 512 | (3,3) | (1,1) | (1,1) | LeakyReLU | — | — |
| BatchNorm2d | 512 | — | — | — | — | 1e − 05 | 0.1 |
| MaxPool | — | (2,1) | (2,1) | 0 | — | — | — |
| Conv2d | 512 | (3,3) | (1,1) | (1,1) | LeakyReLU | — | — |
| BatchNorm2d | 512 | — | — | — | — | 1e − 05 | 0.1 |

| Linear Part | | |
|---|---|---|
| Layer | Input Features | Output Features |
| $Linear_1$ | 1024 | 64 |
| $Linear_2$ | 64 | 64 |

| GRU Part | | | | |
|---|---|---|---|---|
| Layer | Input Size | Hidden Size | # Layers | Dropout |
| RU | 64 | 128 | 2 | 0.25 |
| RU | 256 | 64 | 2 | 0.25 |

| Linear Part | | |
|---|---|---|
| Aye Dense | Input Features | Out Features |
| | 128 | 184 |

**Table 1** Specifications of CNN + GR

| CNN | | | | | | | |
|---|---|---|---|---|---|---|---|
| Layer | # Filters | Kernal size | Stride | Padding | Activation | eps | momentum |

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| Conv2d | 64 | (3,3) | (1,1) | same | ReLU | — | — |
| Conv2d | 64 | (3,3) | (1,1) | same | ReLU | — | — |
| MaxPool | — | (2,2) | (2,2) | Same | — | — | — |
| Conv2d | 128 | (3,3) | (1,1) | Same | ReLU | — | — |
| Conv2d | 128 | (3,3) | (1,1) | Same | ReLU | — | — |
| MaxPool | — | (2,2) | (2,2) | Same | — | — | — |
| Conv2d | 256 | (3,3) | (1,1) | Same | ReLU | — | — |
| Conv2d | 256 | (3,3) | (1,1) | Same | ReLU | — | — |
| Conv2d | 256 | (3,3) | (1,1) | Same | ReLU | — | — |
| MaxPool | — | (2,2) | (2,2) | Same | — | — | — |
| Conv2d | 512 | (3,3) | (1,1) | Same | ReLU | — | — |
| Conv2d | 512 | (3,3) | (1,1) | Same | ReLU | — | — |
| Conv2d | 512 | (3,3) | (1,1) | Same | ReLU | — | — |
| MaxPool | — | (2,2) | (2,2) | Same | — | — | — |
| Conv2d | 512 | (3,3) | (1,1) | Same | ReLU | — | — |
| Conv2d | 512 | (3,3) | (1,1) | Same | ReLU | — | — |
| Conv2d | 512 | (3,3) | (1,1) | Same | ReLU | — | — |
| Linear Part | | | | | | | |
| Layer | Input Features | Output Features | | | | | |
| $\text{Linear}_1$ | 1024 | 64 | | | | | |
| $\text{Linear}_2$ | 64 | 64 | | | | | |
| GRU Part | | | | | | | |
| Layer | Input Size | Hidden Size | # Layers | Dropout | | | |
| GRU | 64 | 128 | 2 | 0.25 | | | |
| Linear Part | | | | | | | |
| Layer | Input Features | Out Features | | | | | |
| Dense | 128 | 184 | | | | | |

**Table 2** Specifications of VGG-16 + GRU

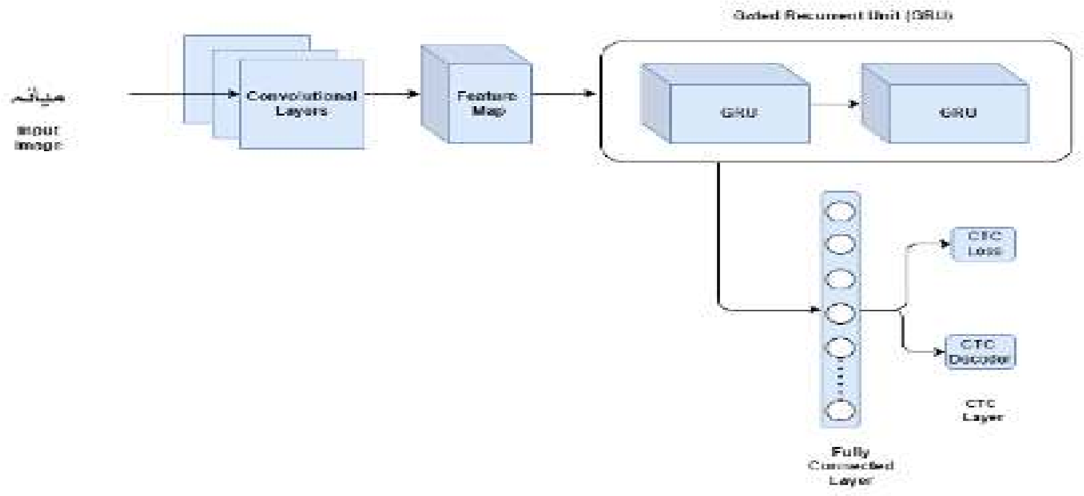


Fig. 9 Proposed Architecture of the CNN + GRU

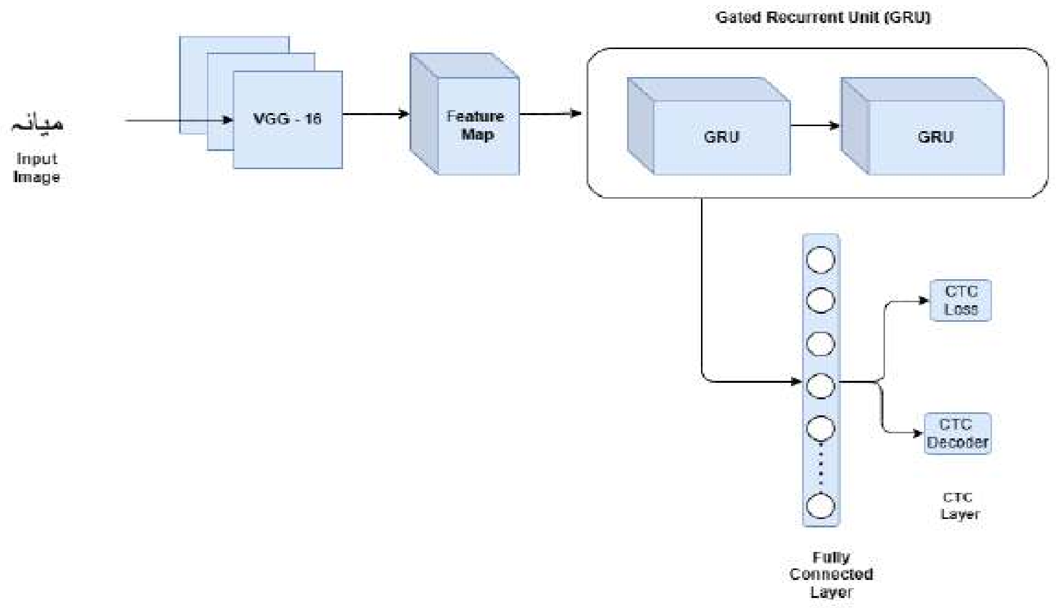


Fig. 10 Proposed Architecture of the VGG16 +GRU

## EXPERIMENTS AND FINDINGS

We conducted extensive tests utilizing deep learning methods to illustrate the model's effectiveness. We begin by describing the performance evaluation indicators and then describe the experimental environment.

## EVALUATION MEASURES FOR PERFORMANCE

To evaluate the effectiveness of deep-learning algorithms in extracting text sequences from images, they are compared against two commonly used performance metrics: word error rate (WER) and character error rate (CER). These metrics provide a quantifiable measure of the accuracy of the algorithm in recognizing text. In both cases, a lower value of these measurements indicates more effectiveness of a strategy, and the opposite is true. These criteria were chosen because they have historically been used to the evaluation of OCR and speech recognition system performance [30, 31]. Such parameters are calculated using the Levenshtein distance [32], which is a measure of how similar two strings are to one another. Using F (C, S) as the character level ground truth, the character level extract predicted by T (C, S), and the Levenshtein distance Leve (G, P) for the image x may be calculated as.

$$\mathrm{CER}_x = \frac{LEVEL_{(C,x)}}{\max\left(F_{(C,x)}, T_{(C,x)}\right)} \tag{7}$$

A distinct collection v N-word or line images is used for testing images. The variable CERv is defined as follows.

$$\mathrm{CER}_v = \frac{1}{N}\sum_{x=0}^{n} CERs \tag{8}$$

Similarly, Word Error Rate (WERx) for image x can be defined as

$$\text{WER}_x = \frac{LEV_{(W,x)}}{\max\left(F_{(W,x)}, T_{(W,x)}\right)} \tag{9}$$

During testing, a set of V is used, where V represents the number of text lines or word images. The word error rate for a specific set v can be represented by the following equation.

$$\text{WER}_v = \frac{1}{N}\sum_{x=0}^{n} WER_x \tag{10}$$

## EXPERIMENTAL CONFIGURATION

PKIT dataset is composed of word, and text-line images. Experiments were done using 120000 text-line and 200000 word level images. The hybrid models CNN+BLSTM and VGG-16+BLSTM were used to perform experiments. The preceding section addresses the design of these models and the ideal settings for the parameters employed in these experiments. The models were implemented using the following specifications:

- Python 3.9.7
- Pytorch 1.13.1
- RAM 32GB
- GPU NVIDIA XP 12GB

In the experiment, 80% of the images were used as the training set, 10% as the validation set, and the remaining 10% as the test set. The model was trained with a batch size of 64 and a learning rate of 0.0005, using the RMSprop optimization algorithm, as suggested by Geoff Hinton in his Coursera class. The training process took place over 300 epochs. Adam was chosen over Stochastic Gradient Descent (SGD) due to its advantages and its common use in training text recognition and image processing models. Table 3 compares the performance of a model under fixed conditions by using equations 8 and 10 to derive CERM and WERM, respectively. Comparing the results, we found that the WER and CER scores for the models were the lowest for word-level lines. Fig 12 shows the training and validation loss of the CNN+GRU model for both word and line level at each epoch. Similarly, 13 shows the training and validation loss of the VGG-16+GRU model for both word and line level. The graph shows the decline in training and validation loss at the end of each epoch.

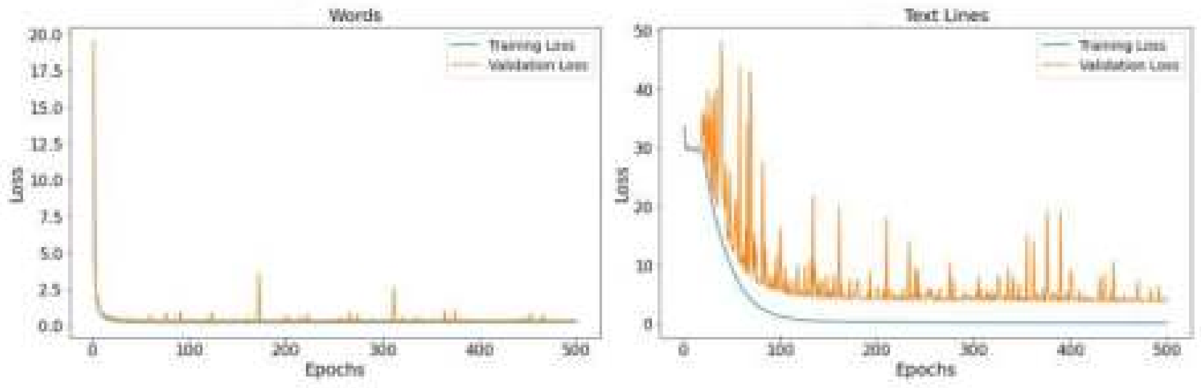


Fig. 11 Loss on training and validation sets at each epoch using CNN+GRU

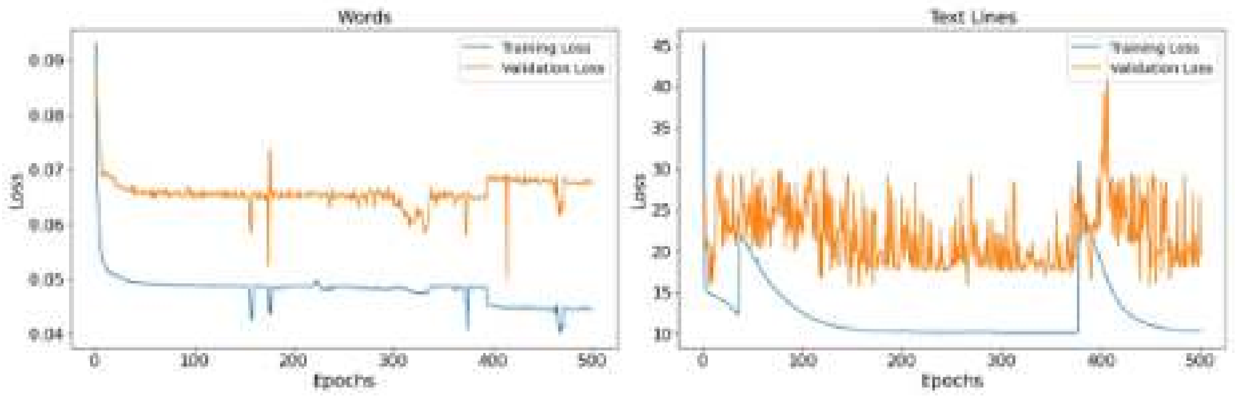


Fig. 12 Loss on training and validation sets at each epoch using VGG-16+GRU

Table 3 Comparison of the performance of the models

| Word Level | | | Text-line level | | |
|---|---|---|---|---|---|
| Model | CER | WER | Model | CER | WER |
| CNN+GRU | 0.067 | 0.12 | CNN+GRU | 0.09 | 0.14 |
| VG16+GRU | 0.042 | 0.06 | VGG16+GRU | 0.07 | 0.10 |

## CONCLUSION

In this study, we proposed a novel dataset of printed Kashmiri pictures at the line and word level. The synthetic technique is utilized to produce images along with their associated ground truth. Furthermore, our proposed dataset was evaluated using deep learning methods for optical character recognition (OCR). Experimental results show that the deep learning model extended well to the proposed dataset, enabling us to get exceptional WER and CER results.

In spite of our development of the first-ever Kashmiri OCR dataset, we would want to see more data donated to our dataset in terms of volume and diversity. Additionally, image rendering characteristics such as text size, background color, and foreground color may be changed. Additionally, various strategies for data augmentation may be employed to train our model. Other future research objectives include the development of even more performing end-to-end OCR application Kashmiri script via the use of other machine learning models like ensembled and attention-based models. Finally, our suggested approach may be used for the development of end-to-end OCR applications for Urdu, Punjabi, Hindi, and Sindh.

## REFERENCES

[1] Davis, R., Lyall, J.: Recognition of handwritten characters — a review. Image and Vision Computing 4(4), 208–218 (1986). https://doi.org/10. 1016/0262-8856(86)90048-X

[2] Plamondon, R., Srihari, S.N.: On-line and off-line handwriting recognition: A comprehensive survey. IEEE Trans. Pattern Anal. Mach. Intell. 22, 63–84 (2000)

[3] Isheawy, N.A.M., Hasan, H.: Optical character recognition (ocr) system. IOSR Journal of Computer Engineering (IOSR-JCE), e-ISSN, 2278–0661 (2015)

[4] Gon, calves, G.R., Diniz, M.A., Laroca, R., Menotti, D., Schwartz, W.R.: Real-time automatic license plate recognition through deep multi-task networks. In: 2018 31st SIBGRAPI Conference on Graphics, Patterns and Images (SIBGRAPI), pp. 110–117 (2018). IEEE

[5] Qadri, M.T., Asif, M.: Automatic number plate recognition system for vehicle identification using optical character recognition. In: 2009 International Conference on Education Technology and Computer, pp. 335–338 (2009). IEEE

[6] Fujisawa, H.: Forty years of research in character and document recognition—an industrial perspective. Pattern Recognition 41(8), 2435–2446 (2008)

[7] Mohamed, M., Gader, P.: Handwritten word recognition using segmentation-free hidden markov modeling and segmentation-based dynamic programming techniques. IEEE transactions on pattern analysis and machine intelligence 18(5), 548–554 (1996)

[8] Tappert, C.C., Suen, C.Y., Wakahara, T.: The state of the art in online handwriting recognition. IEEE Transactions on pattern analysis and machine intelligence 12(8), 787–808 (1990)
[9] Breuel, T.M., Ul-Hasan, A., Al-Azawi, M.A., Shafait, F.: Highperformance ocr for printed english and fraktur using lstm networks. In: 2013 12th International Conference on Document Analysis and Recognition, pp. 683–687 (2013). IEEE

[10] Cesar, M., Shinghal, R.: An algorithm for segmenting handwritten postal codes. International journal of man-machine studies 33(1), 63–80 (1990)

[11] Cheriet, M., Huang, Y.S., Suen, C.Y.: Background region-based algorithm for the segmentation of connected digits. In: 11th IAPR International Conference on Pattern Recognition. Vol. II. Conference B: Pattern Recognition Methodology and Systems, vol. 1, pp. 619–620 (1992). IEEE Computer Society

[12] Doetsch, P., Kozielski, M., Ney, H.: Fast and robust training of recurrentneural networks for offline handwriting recognition. In: 2014 14th International Conference on Frontiers in Handwriting Recognition, pp. 279–284 (2014). IEEE

[13] Natarajan, P., Bazzi, I., Lu, Z., Makhoul, J., Scwhartz, R.: Robust ocr of degraded documents. In: Proceedings of the Fifth International Conference on Document Analysis and Recognition. ICDAR'99 (Cat. No. PR00318), pp. 357–361 (1999). IEEE

[14] Graves, A., Schmidhuber, J.: Offline handwriting recognition with multidimensional recurrent neural networks. Advances in neural information processing systems 21 (2008)

[15] Ashwin, T., Sastry, P.: A font and size-independent ocr system for printed kannada documents using support vector machines. Sadhana 27(1), 35–58 (2002)

[16] Bashir, M., Goyal, V., Giri, K.J.: Challenges in recognition of Kashmiri script. In: Singh, P.K., Singh, Y., Kolekar, M.H., Kar, A.K., Gon, calves, P.J.S. (eds.) Recent Innovations in Computing, pp. 33–43. Springer, Singapore (2022)

[17] Bashir, R., Quadri, S.M.K.: Identification of kashmiri script in a bilingual document image. 2013 IEEE Second International Conference on Image Information Processing (ICIIP-2013), 575–579 (2013)

[18] Graves, A., Schmidhuber, J.: Offline arabic handwriting recognition with multidimensional recurrent neural networks, pp. 545–552 (2008). https: //doi.org/10.1007/978-1-4471-4072-6 12

[19] Pal, A.: Bengali handwritten numeric character recognition using denoising autoencoders. In: 2015 IEEE International Conference on Engineering and Technology (ICETECH), pp. 1–6 (2015). IEEE

[20] Ahmad, I., Wang, X., Li, R., Rasheed, S.: Offline urdu nastaleeq optical character recognition based on stacked denoising autoencoder. China Communications 14(1), 146–157 (2017)

[21] Naz, S., Umar, A.I., Ahmad, R., Siddiqi, I., Ahmed, S.B., Razzak, M.I., Shafait, F.: Urdu nastaliq recognition using convolutional–recursive deep learning. Neurocomputing 243, 80–87 (2017)

[22] Addakiri, K., Bahaj, M.: Article: On-line handwritten arabic character recognition using artificial neural network. International Journal of Computer Applications 55(13), 42–46 (2012). Full text available

[23] Osman, H., Zaghw, K., Hazem, M., Elsehely, S.: An efficient languageindependent multi-font OCR for arabic script. CoRR abs/2009.09115 (2020) 2009.09115

[24] Fasha, M., Hammo, B.H., Obeid, N., Widian, J.: A hybrid deep learning model for arabic text recognition. ArXiv abs/2009.01987 (2020)

[25] Altwaijry, N., Al-Turaiki, I.: Arabic handwriting recognition system using convolutional neural network. Neural Computing and Applications 33 (2021). https://doi.org/10.1007/s00521-020-05070-8

[26] Mushtaq, M.M.M.K.M.K. Faisel, Sing, S.: Urdudeepnet: offline handwritten urdu character recognition using deep neural network. Neural Computing and Applications 33 (2021)

[27] Qurat-ul-Ain Akram, F.A.S.U.S.H.S.S. Anneta Niazi: A comprehensive image dataset of urdu nastalique document images, 81–88 (2016)

[28] 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. IEEE transactions on pattern analysis and machine intelligence 39(11), 2298–2304 (2016)

[29] Simonyan, K., Zisserman, A.: Very Deep Convolutional Networks for Large-Scale Image Recognition. arXiv (2014). https://doi.org/10.48550/ ARXIV.1409.1556. https://arxiv.org/abs/1409.1556

[30] Wei, T.C., Sheikh, U., Ab Rahman, A.A.-H.: Improved optical character recognition with deep neural network. In: 2018 IEEE 14th International Colloquium on Signal Processing & Its Applications (CSPA), pp. 245–249 (2018). IEEE

[31] Paul, D., Chaudhuri, B.B.: A BLSTM network for printed bengali OCR system with high accuracy. CoRR abs/1908.08674 (2019) 1908.08674

[32] Levenshtein, V.I., et al.: Binary codes capable of correcting deletions, insertions, and reversals. In: Soviet Physics Doklady, vol. 10, pp. 707–710 (1966). Soviet Union