I.J. Image, Graphics and Signal Processing, **2025, 6, 133-150**
Published Online on December 8, 2025 by MECS Press (http://www.mecs-press.org/)
DOI: 10.5815/ijigsp.2025.06.08

E-Chars74k: An Extended Scene Character Dataset with Augmentation Insights and Benchmarks

Payel Sengupta*
Department of Computer Science and Engineering, Brainware University, Barasat, West Bengal 700125, India
E-mail: payel9433@gmail.com
ORCID iD: https://orcid.org/0000-0003-3981-5971
*Corresponding Author

Tauseef Khan
School of Computer Science and Engineering, VIT-AP University, Amaravati, Andhra Pradesh, 522237, India
Email: tauseef.khan@vitap.ac.in
ORCID iD: https://orcid.org/0000-0002-3359-9967

Ayatullah Faruk Mollah
Department of Computer Science and Engineering, Aliah University, Kolkata 700160, India
Email: afmollah@aliah.ac.in
ORCID iD: https://orcid.org/0000-0002-3445-7469

Received: 05 December, 2024; Revised: 11 May, 2025; Accepted: 16 October, 2025; Published: 08 December, 2025

Abstract: Semantic understanding of camera-captured scene text images is an important problem in computer vision. Scene character recognition is the pivotal task in this problem, and deep learning is now-a-days the most prospective approach. However, limited sample-size of scene character datasets appear to be a major hindrance for training deep networks. In this paper, we present (i) various augmentation techniques for increasing the sample size of such datasets along with associated insights, (ii) an extended version of the popular Chars74k dataset (herein referred to as E-Chars74k), and (iii) the benchmark performance on the developed E-Chars74k dataset. Experiments on various sets of data such as digits, alphabets and their combination, belonging to the usual as well as wild scenarios, clearly reflect significant performance gain (20%-30% increase in scene character recognition accuracy). It is noteworthy to mention that in all these experiments, a deep convolutional neural network powered with two conv-pool pairs is trained with the uniform training test partition to foster comparison on equal bench.

Index Terms: Scene Character Recognition, Deep Learning, CNN, Augmentation, Chars74k Dataset

1. Introduction

Character recognition is an extensively studied problem in pattern recognition and image processing. However, scene character recognition is a relatively new problem that emerged with the unprecedented progress in scene text detection with the advent of deep learning [1]. Unlike conventional character recognition, classification of dissected scene characters is significantly challenging due to complex foreground-background, partial overlapping, and variation in font characteristics, translated position, and different degradation as well as distortion issues, some of which are reflected in Fig. 1. Traditional Optical Character Recognition (OCR) systems usually do not take care of these issues as they have focused mainly on optically scanned characters of document images obtained with flatbed scanners. It is noteworthy that scene character recognition is an essential phase in scene text reading systems. Due to the non-contact and freehold nature of built-in digital cameras, scene text images are mostly camera-captured images that often carry meaningful information like address, phone number, description, etc. [2]. Automatic retrieval of textual information from such images may ease human lifestyle. Some of its applications include cell phone navigator, phone number collection, scene text translation, etc.

Considering the diversity of challenges involved in scene character recognition, it is a standalone research task wherein more attention from the research fraternity is imperative. According to a study [3], scene character recognition methods may be grouped into (i) direct recognition [5], (ii) word image classification [6], (iii) end-to-end recognition through text detection and recognition [7], (iv) sequence-based recognition [8], and (v) hybrid recognition [9]. While the problem of scene character recognition is nascent, it has been mainly studied as part of text detection and recognition methods. Only a few works may be found to focus on the recognition of individual scene characters dissected from scene word images, and in general, these methods have analyzed their performance separately on digits, alphabets, and a combination of them. Though many scene word datasets [10, 11, 12, 13] are available, scene character datasets [14,15] are not available in plenty. As deep learning methods often yield extraordinary performance compared to classical hand-crafted feature-based shallow classifiers, such methods are increasingly applied in many image classification tasks [16]. Deep learning methods usually require massive amounts of data for proper training and prediction. However, the size of the scene character dataset(s) must be increased for deep networks.

In the ICDAR 2003 dataset [14], there are only 6,185 character-images for training and 5,430 characters for testing. However, these samples are not grouped class-wise. Chars74k [15], on the other hand, is a composite dataset of diversified characters. Without dividing them into training and test sets, it contains 12,503 scene character images, including 887 digits and 11,616 alphabets (uppercase A and lowercase a-z). Here, the total number of classes is 62, i.e., 10 for digits, 26 for uppercase letters, and 26 for lowercase letters. Further, these 12,503 images are divided into two sets, viz. (a) good images and (b) bad images, to facilitate performance analysis in order of complexity. In the excellent set, class 'l' contains the lowest number of samples, i.e. 33 and class 'A' includes the highest number of samples, i.e. 558. In contrast, in the bad set, the range of samples per class is even shorter, i.e., 2-360 ('q' contains only two samples, and 'A' includes 360 samples). Thus, it is evident that these many samples are insufficient for appropriate training of deep networks. Moreover, the number of samples per class varies significantly, with some classes having a minimal number of samples, which may lead to undertraining and imbalanced training. Therefore, augmenting datasets is a popular and practical approach to increase the number of samples and improve the performance of deep learning frameworks.

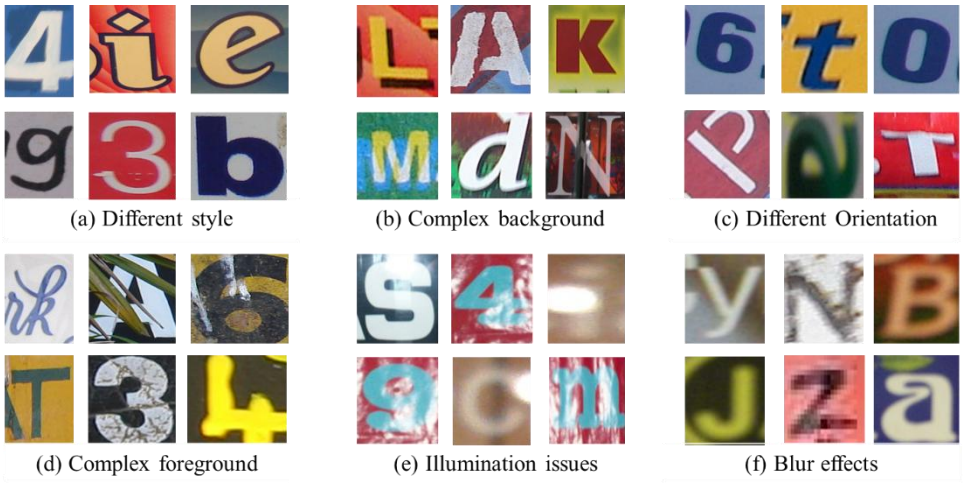


Fig. 1. Sample scene character images from the Chars74k original dataset, bearing different challenges (Characters of different styles, backgrounds, orientations, foreground, illumination, and degradation issues are shown).

In this paper, we have presented (i) an overview of commonly used augmentation techniques along with insightful discussion, (ii) preparation of an extended version of Chars74k through augmentation, (iii) a deep-learning inspired method for character recognition, and (iv) performance study in order of complexity of scene characters at multiple levels viz. digits, alphabets and their combination for good set as well as bad set. Augmentation techniques and related discussion may provide deeper insights to readers about the selection and application of appropriate augmentation techniques for a given image classification problem. As the samples of the original Chars74k(Scene) dataset are not divided into training and test sets, some methods randomly divide them and report their results that cannot be compared on an even bench as the pre-requisite of a separate and unique test set still needs to be fulfilled. On the contrary, the augmented dataset, i.e., E-Chars74k, has an individual training and test set of samples for both the good and the bad set. It is also released for more significant usage and performance benchmarking at an even scale. Obtained results on different subsets of this dataset reflect the benefit of augmentation, which may serve as the initial benchmark performance, and researchers are encouraged to outperform. The organization of the paper is as follows: Related works are discussed in Section 2, an overview of augmentation techniques is included in Section 3, and preparation of E-Chars74k is discussed in Section 4. architecture of the employed deep network is given in Section 5, experimental results are presented in Section 6, and finally, in Section 7, a conclusion is drawn, and the scope of future work is outlined.

2. Related Works and Motivation

Traditional character recognition works primarily focused on optically scanned well-segmented characters, but studies on scene character recognition are not available in plenty [3]. Moreover, research in scene character recognition reported so far is mainly wrapped with end-to-end text detection and recognition [4]. There are only a few works focused on dissected scene text reading. Until recently, scene character recognition methods heavily relied on classical hand-crafted features [17]. De Campos et al. [15] applied traditional features such as shape contexts, geometric blur, scale-invariant feature transform (SIFT), spin image, the maximum response of filters, and patch descriptor followed by KNN and SVM classifiers. Experiments were conducted on English and Kannada characters with scenes in nature and 62 classes. Sengupta et al. [18] developed a recognition approach using a histogram of oriented gradients (HOG) features where only English character images of Chars74k [15] were converted into a grayscale, histogram equalization was applied, and classification was done with multiple classifiers such as Naive Bayes, MLP, KNN, SVM, and Random Forest. Chekov et al. [19] applied two local feature descriptors, i.e., SIFT and HOG, to recognize segmented characters of Chars74k and ICDAR2003 datasets. Lee et al. [20] reported a method for scene text recognition where the proposed feature polling is applied to divide scene characters into sub-regions within a multi-class classification framework. Using datasets like Chars74k, ICDAR2003, ICDAR2011, and SVT, they found that the algorithm performed well even when there were considerable variations in the foreground and background of natural images.

Sengupta et al. [21] reported a scene character classification method using morphology-based filtering, HOG descriptors, and different classifiers. It was applied to the scene characters of the Chars74k dataset, and SVM was found to outperform other classifiers. Ali et al. [22] proposed a holistic character recognition method that can be applied to scene characters when they occur in a rotated position. It aimed to overcome local feature dependency, and hence, this method produces reasonably good results even for rotated characters. This method is applied to various datasets like ICDAR2003, Chars74k-Image, and Chars74k-Font and has been found to yield better results than local features such as HOG. Neumann et al. [23] proposed a novel method for scene character recognition. It is based on Maximally Stable Extremal Regions (MSERs) and has been shown to overcome the problem of different illumination and geometry in scene text recognition. On the Chars74k dataset, this method produced 18% better results than the state-of-the-art. Coates et al. [24] developed a scene text detection and character recognition method by automatically selecting features with unsupervised methods such as variant k-means clustering followed by a linear SVM classifier. They reported its performance on the ICDAR2003 dataset.

Though, some of these methods produced reasonably high recognition accuracies on small datasets, such methods need to be more general and robust to a great extent. Due to their unprecedented performance, scene character recognition methods rely heavily on deep learning. Different convolutional neural network (CNN) based deep learning architectures are used for automatic feature extraction. Thus, the prediction accuracy obtained is much higher than that of classical hand-crafted methods. An essential requirement of such methods is the large dataset size for training. Though, large-sized scene character datasets are not available to our knowledge, naive deep models have been applied, and encouraging results have been obtained. For instance, Arroyo-Perez et al. [25] proposed a method for automatic license plate recognition of Colombian automobiles using CNN, wherein the Chars74k dataset is used for training. Only digits and upper-case letters have been chosen for this training because Colombian car license plates have only two characters in uppercase letters followed by some digits. Chandio et al. [26] proposed a network architecture based on multi-level feature fusion (MLFF) and multi-scale feature aggregation (MSFA) to recognize Urdu characters from natural images. The proposed neural network model is applied to English Chars74k datasets. Both cases, i.e. MLFF and MSFA, demonstrated better results than other methods. Lin et al. [27] reported a novel approach to recognizing scene characters using local binary pattern networks called LBPNet, which works on characters' strokes and distinct outlines. This method is applied to benchmark datasets like MNIST, SVHN, ICDAR2003, and Chars74k.

Abdali et al. [28] proposed a CNN architecture for both characters and digit recognition on natural images. Random data augmentation is done on Chars74k and EMINST to train the CNN model and is applied to ICDAR2003 (non-scene dataset). Zhang et al. [29] developed an encoder-based method named bilateral convolutional activation encoded with Fisher vectors (BCA-FV). Convolutional activation descriptors are extracted from convolutional maps and then applied to bilateral convolutional activation maps to describe the fundamental relation between the information of basic spatial structure and the response of convolutional activation. This method is verified on Chars74k and ICDAR2003 benchmark datasets. Driss et al. [30] applied MLP and CNN to the Chars74k dataset. MLP is used with a Unified character descriptor, and CNN is applied with LeNet-5. As a result, it has been observed that real-time CNN is more efficient than MLP to classify dissected characters. Zhang et al. [31] developed consecutive convolutional activations (CCA) to recognize scene characters. Using conventional layers, CCA integrates high-level and low-level patterns into global decisions by learning character structures. This method is applied to English (Chars74k and ICDAR2003) and Chinese scene character datasets. Neoh et al. [32] proposed a deep learning-based architecture for vehicle plate recognition by training with natural and computer font images collected from Chars74k and CIFAR10 datasets. It can recognize different digits and alphabets. Ray et al. [33] proposed a deep learning-based approach. It is based on automated feature extraction, which gives better accuracy than the traditional hand-crafted method. Results are compared with different training algorithms like multi-layer perceptron, logistic regression, De-noising Score Matching,

and Contrastive Divergence. These algorithms are applied to 4 benchmark datasets: Chars74k Kannada, Chars74k English, SVT-CHAR dataset, and ICDAR 2003 dataset.

Zhang et al. [34] developed a deeper convolutional neural network with 15 layers and 3X3 tiny receptive fields to extract better features from scene character images. All experiments are done on Chars74k, SVT, and ICDAR2003 datasets. Ali et al. [35] developed a holistic solution for scene character recognition using a 3-mode tensor. Every training tensor is decomposed into a liner superposition of 'k' rank-1 matrices, whereas rank-1 matrices are the spanning solution of the character classes. Chars74k, SVT-CHAR, and ICDAR2003 datasets were used to generate experimental results. Li et al. [36] proposed a CNN architecture based on a compact binarized neural network (CBNN). This architecture has overcome the drawback of a binarized neural network (BNN) based on a hardware framework. The essential components of CBNN are bit-level data pruning and bit-level sensitivity analysis of each character class. Cifar-10, Char74K, GTSRB, and Cifar-10 benchmark datasets quantify the proposed architecture results. Wang et al. [37] proposed discriminative CNN-based architecture for recognition of scene characters images. This architecture considers convolutional activations as local descriptors of the scene's character images. The proposed method is applied to benchmark datasets ICDAR03, IIIT5K, and Chars74k.

However, the above experiments are applied to ICDAR 2003 and Chars74k datasets. In this paper, the Chars74k dataset has been chosen for the deep learning-based recognition approach because most of the work done on the Chars74k dataset is done in different ways, and state-of-the-art results must be appropriately reported due to other parameters being taken for experiments. In the case of the Chars74k dataset, two types of images are good and bad images. All experimental results are reported with accuracy but can't specify in which dataset (good and bad) method has been applied, and most of the deep learning-based character recognition was done on the original Chars74k dataset without augmentation. This work has proved that after augmentation of the Chars74k dataset, it gives more efficient results than the original dataset. A new augmented English character dataset has been developed with 1,92,558 dissected character images (digits and alphabets). Many datasets have been modified to generate acceptable results for deep learning models. Another object is that images are arranged in class among two dissected character datasets (ICDAR2003 and Chars74k) in the Chars74k dataset, which is helpful for labeling in deep learning.

3. Overview of Augmentation Techniques

Deep learning models usually require feeding massive amounts of data for proper training [1]. Datasets of small size are reasonably acceptable for performance analysis of many traditional methods. The advent of deep learning in recent times necessitates the development of significantly large datasets. However, collecting sufficient data and preparing proper annotations are not straightforward; such datasets are only available in some fields. To overcome this problem of the unavailability of large datasets, an effective and convenient way is to apply different data augmentation techniques for deep learning experiments. Augmentation increases the number of samples using various transformations to an original image. Data augmentation can fulfill the bulky data requirement for training deep learning models. On the other hand, it may empower the design of robust deep networks. The basic idea is that by increasing the size of the training dataset, a deep neural model can be better trained, and prediction performance can be improved significantly [38,39]. It is noteworthy to mention that test samples need not be augmented. The core idea of augmentation lies in the training of deep networks. This section presents an overview of different image augmentation techniques with applicable datasets and basic guidelines for selecting appropriate augmentation techniques for a given image classification problem.

3.1 Augmentation Techniques

Before the advent of deep learning, augmentation was not of substantial interest and, hence, was confined to fundamental transformations such as color space, horizontal flipping, and random cropping. Nowadays, driven by the need for a large volume of training data for many deep learning-based approaches, a wide range of augmentation techniques [40] has come into the picture, briefly discussed below.

a) *Flipping:* This geometric transformation is one of the earliest data augmentation methods. Horizontal flipping is commonly used rather than vertical flipping. In some robust datasets like ImageNet and CIFAR-10, this flipping geometric transformation is used for image augmentation. However, as it is not a label-preserving transformation, it cannot be used for specific problems like character classification.
b) *Cropping*: Cropping is another augmentation process for images. Regions of different dimensions may be cropped from the original image to obtain transformed samples. Random cropping is one of the possible ways to cut from the original image, unlike translations dynamically. A subtle difference between translation and cropping is that translations usually preserve the spatial dimension, whereas cropping always reduces the actual size of input images.
c) *Rotation*: The image is rotated in a clockwise or anticlockwise direction by an angle between 10 and 359 degrees. Slight rotation may be decisive for digits and character recognition problems. Chars74k, ICDAR2003, and MNIST are examples of such datasets. In case of rotation by a large angle, the original labels of images may not remain preserved in specific problems.

d) *Translation*: To avoid positional bias in images, different sliding transformations (up, down, left, right) may be beneficial for image recognition. This type of augmentation applies to many kinds of datasets. Padding, wherever necessary, may be used to preserve the spatial dimension of image samples.
e) *Noise injection*: Noise injection means using Gaussian noise and Salt and Pepper noise randomly values are injected. Gaussian noise injection type of augmentation is done by Moreno-Barea et al. [41] and applied to datasets in the UCI repository [42]. Noise injection augmentation helps to train CNN with more classification features.
f) *Colour space transformation*: Scene images are encoded with 3 (R, G, B) stacked matrices, and each image size has height x. R, G, and B pixel values represent matrices. This photometric transformation is constructive for the recognition of images. This color distribution can reduce the lighting challenges faced by testing data.
g) *Kernel filters*: Blurring and sharpening images with kernel filters are well-known techniques for image augmentation. Gaussian blur is often applied in kernel-based filtering. Critics of kernel filter-based augmentation may argue that this is equivalent to the internal CNN mechanism, done layer-by-layer. Nevertheless, it may be beneficial, as reflected in the literature. For instance, Kang et al. [43] have applied a kernel filter on the CIFAR-10 dataset for use in ResNet architecture [44].
h) *Mixing images*: Another powerful data augmentation is mixing two or more images by averaging pixel intensities. Images obtained this way are found to positively contribute to the training of deep networks and produce better results. Applying this augmentation strategy to the CIFAR-10 dataset, Ionue et al. [45] have reduced the error rate from 8.22% to 6.93%.
i) *Random erasing*: Some parts of the image are discarded in this type of augmentation. It is applied by random selection and removal of $x \times y$ patch of an input image. After applying this augmentation method, the error rate is reduced from 5.17% to 4.30% on the CIFAR-10 dataset. Terrance et al. [46] proposed cutting out regularization for random erasing. A possible drawback of this technique is that removing a large patch may not permanently preserve the sample's label.
j) *Combined augmentation*: A combination of different transformations like random erasing, flipping, cropping, etc., is one of the best ways to increase the size of datasets. That will help train deep learning architecture.

3.2 Selection of Appropriate Augmentation Techniques

As evident in Section 3.1, not all augmentation techniques suit all images. A significant consideration in selecting appropriate augmentation techniques is the preservation of labels. A few guidelines are important while deciding the augmentation technique [47] based on the problem you are trying to solve. Here is a summary of these guidelines:

(i) The first step in any model-building process is to ensure that the size of our input matches what the model expects. We also have to make sure that the size of all the images should be similar. For this, we can resize our images to the appropriate size.
(ii) In a classification problem, if the number of data samples is relatively less, then different augmentation techniques like image rotation, image noising, flipping, shift, gamma correction, sigmoid correction, rotation, etc., are used. All these operations apply to classification problems where the location of objects in the image does not matter.
(iii) Normalizing image pixel values is always an excellent strategy to ensure better and faster convergence of the model. If the model has specific requirements, we must pre-process the images per the model's requirement.

4. Preparation of E-Chars74k Dataset

The original Chars74k dataset was developed by De Campos et al. [15] in 2009. This dataset includes symbols of English and the Kannada language. This dataset has included natural, hand-printed font character images. Among them, 3410 English and 16425 Kannada character images are hand printed using PC, 62992 are English character images with 254 different fonts and four styles (regular, bold, italic, and bold + italic), and 12,505 scene character images, including digits and alphabets. As the prime focus of this work is scene character recognition, only scene character images of digits (0-9), uppercase characters (A-Z), and lower-case characters (a-z) are chosen. Among these 12,503 scene character images, there are only 717 digits and 9,334 alphabets (uppercase and lowercase). Thus, considering an alphanumeric classification problem, the total number of classes becomes 62 (10 classes for digits, 26 classes for uppercase letters, and 26 classes for lowercase letters). Depending on the level of complexity, these 12,503 images have been further grouped as (a) good images (7,705) and (b) bad images (4,798), both having 62 classes. Sample images of the naïve good set and the lousy set are shown in Fig. 2.

Challenges. It may be noted from Fig.2 that the images of the bad set could be more straightforward to visualize and recognize. Some noteworthy challenges in such scene character recognition include different character font characteristics, blur, illumination issues, geometric distortion, arbitrary orientation, the unwanted appearance of extra portions with character strokes, complex backgrounds, etc. Moreover, many characters need to be appropriately dissected. Additionally, it may be noticed that there needs to be more samples in some classes. Utilizing such a challenging dataset in the investigation and development of practically effective scene character recognition methods is

undoubtedly worth it. However, as the size of this dataset is smaller, deep learning methods remain undertrained, causing unsatisfactory performance. Traditional approaches, however, have utilized this dataset to train and predict dissected scene characters. However, the performance of such methods could not be compared on even platforms since no separate and unique training or test sets are available.

(a) (b)

Fig. 2. Sample images from the original Chars74k dataset (a) Relatively good images with different characteristics, and (b) bad images with blur, uneven lightening, heterogeneous background, and other complexities.

Training Test Partition. The fundamental idea to prepare the E-Chars74k dataset is to partition the Chars74k dataset into training (80%) and test (20%) sets, at first, and then, to enlarge only the training data by 18 different augmentation techniques. Augmented images of the E-Chars74k dataset (https://doi.org/10.6084/m9.figshare.17294069) are shown in Fig. 3 and Fig. 4. The number of samples present in each class of Chars74k and E-Chars74k are mentioned in Table 1. It is important to note that the test set is kept intact in order to avoid any bias towards training of any classifier or deep network. In case of augmentation of a whole dataset and its subsequent partition, prediction becomes biased lest a sample get predicted with a model trained with its own augmented version, which is totally unacceptable. Moreover, a distinct and original test set is always preferred for benchmark performance comparision.

Augmentation. The above-mentioned morphological operations are briefly described below, one by one. Below, we discuss the preparation of the E-Chars74k dataset, an extended version of the Chars74k scene character dataset, by applying some suitable augmentation techniques following the insights of the augmentation overview presented in Section 0. Detailed descriptions of the original Chars74k dataset and augmented Chars74k dataset are described below. Various morphological operations such as erosion, dilation, opening, closing, and transformations such as Gaussian filter, Median filter, rotation positive (anticlockwise rotation 5 degrees), rotation negative (clockwise 5 degrees), introducing noise to images, histogram equalization, contrast enhancement, contrast diminution, gamma correction, logarithm correction, color space, shearing, etc. have been applied.

After a brief study of the scene character dataset, we concluded that this dataset is the only segmented scene character dataset of Roman text. It can be further stated that this work is performed in Roman characters only due to various languages having many challenges. The main challenge of this dataset is that after augmentation of the original dataset, the number of character images increases automatically and gives a satisfactory result than the recognition accuracy of the original Chars74k dataset using CNN and traditional classifier.

i. **Erosion:** The basic idea of erosion is just like soil erosion; it erodes the boundaries of the foreground objects (Always try to keep the foreground white). We have chosen a 5x5 size kernel in our dataset preparation as a parameter.
ii. **Dilation:** It is just the opposite of erosion. Here, a pixel element is '1' if at least one pixel under the kernel is '1'. So, it increases the image's white region or the foreground object's size. Usually, in cases like noise removal, erosion is followed by dilation. Because erosion removes white noises, but it also shrinks our objects. So, we dilate it. Since the noise is gone, they won't return, but our object area increases. It is also helpful in joining broken parts of an object. We have chosen a 5x5 size kernel in our dataset preparation as a parameter.
iii. **Opening:** Opening is just another name of erosion followed by dilation. It helps remove noise. We have chosen a 5x5 size kernel in our dataset preparation as a parameter.
iv. **Closing:** Closing is the reverse of Opening and dilation followed by Erosion. It is useful in closing small holes inside the foreground objects or minor black points on the object. Here, we have considered a 5x5 kernel for our dataset.
v. **Gaussian Filtering**: In this approach, instead of a box filter consisting of equal filter coefficients, a Gaussian kernel is used. Kernel size 7x7 has been used for our dataset augmentation.
vi. **Median Filtering:** The median of all the pixels under the kernel window and the central pixel is replaced with this median value. This is highly effective in removing salt-and-pepper noise. The kernel size must be a positive odd integer. Here in this experiment, we have considered a 5x5 size kernel.

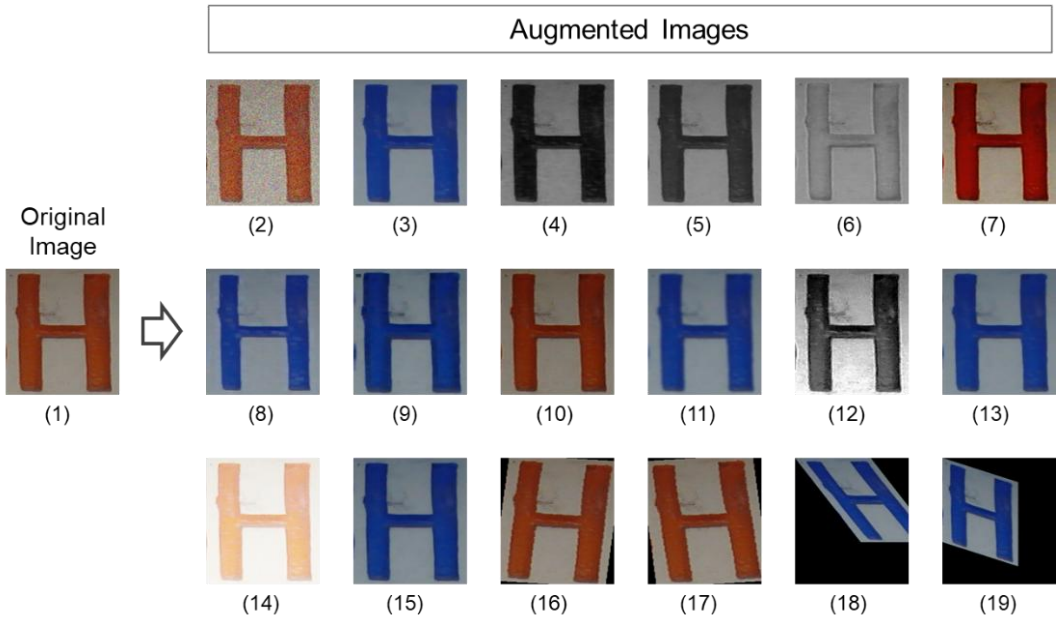
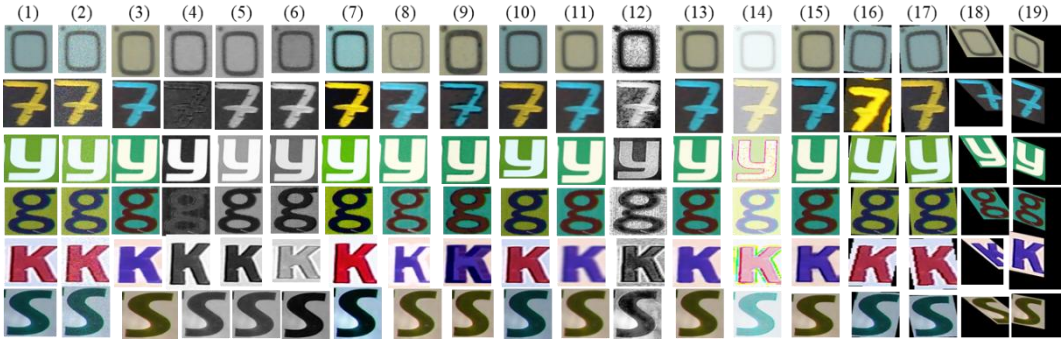


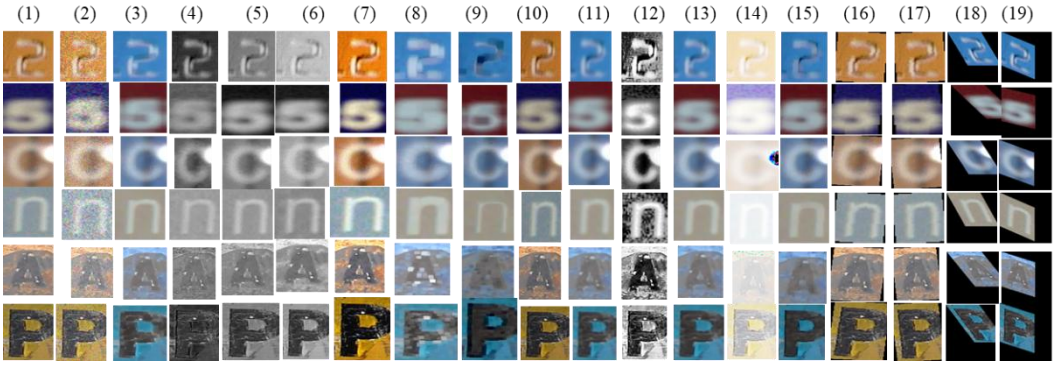
Fig. 3. An original image and the augmented images obtained with 18 different types of augmentation strategies. (1) Sample original Chars74k image, (2) Noise addition, (3) Closing, (4) Color space conversion Blue, (5) Color space conversion Red, (6) Color space conversion Green, (7) Contrast stretching, (8) Dilation, (9) Erosion, (10) Gamma correction, (11) Gaussian filtering, (12) Histogram equalization, (13) Image median, (14) Logarithm correction, (15) Opening, (16) Rotation operation (clockwise), (17) Rotation operation (anti-clockwise), (18) Shear transformation (X-origin), and (19) Shear transformation (Y-origin).

vii. **Rotation:** A positive value of the angle rotates the image in an anticlockwise direction, while a negative value of the angle rotates the image in a clockwise direction. Rotation operation, as the name suggests, turns the image by a specified degree. In our dataset preparation, we rotated our sample images in 5 degrees clockwise and anticlockwise.
viii. **Adding noise to images:** Image noising is an important augmentation step that allows our model to learn how to separate signal from noise in an image. This also makes the model more robust to changes in the input. In our experiment, Gaussian noise is applied.
ix. **Histogram equalization:** An intensity distribution of an image is represented by histogram equalization. This method increases the global contrast value of an image. Histogram equalization is applied for escalation of lower contrast to higher contrast.
x. **Contrast stretching:** Contrast stretching is an Image Enhancement method that attempts to improve an image by stretching the range of intensity values. Max-min contrast starching has been taken for this experiment.
xi. **Gamma correction**: It controls the overall brightness of an image. Not properly corrected images can look either bleached out or too dark. In this experiment, the gamma value 1.2 is selected.
xii. **Logarithmic correction**: After it is applied to a digital character image, the darker intensity values are given brighter values, thus making the details present in darker or gray areas of the image more visible to human eyes.
xiii. **Color space conversion (R/G/B):** Color space conversion is another data augmentation technique with more than one color mode. The most used color space conversion is RGB color conversion, i.e., three color coordinates have made it with red, blue, green, and blue. It is used to convert one color space to another. All intensity values after RGB color conversion is within range from 0 to 255.
xiv. **Shear transformation:** Shear transformation is a linear mapping that displaces each point in a static direction by an amount related to its signed distance from the line, which is parallel to that direction and goes through the origin. In our experiment, shear transformation about the X axis and about Y are used for data augmentation.

Different augmented samples of Chars74k (good set and bad set) are shown in Fig. 3 and Fig. 4. Graphical representations of sample distribution are shown in Fig. 5. Number of training samples and test samples are quantified in Table 1.



(a) Samples from E-Chars74k (Good set)



(b) Samples from E-Chars74k (Bad set)

Fig. 4. Original image (1) and augmented samples (2-18) technique applied on (a) good images and (b) bad images of E-Chars74k dataset, where (1) Sample of original Chars74k image (2) - (18) Sample of augmented images.

Table 1. Number of training and test images in Chars74k and the E-Chars74k dataset (Digits, Alphabets, and combination of digits and alphabets)

Dataset	Subset	Group	Training Set	Test Set	Total
Chars74k	Good	Digits	476	119	595
		Alphabets	5,688	1,422	7,110
		Digits+Alphabets	6,164	1,541	7,705
	Bad	Digits	234	59	292
		Alphabets	3,605	921	4,506
		Digits+Alphabets	3,839	960	4,798
E-Chars74k	Good	Digits	9,044	119	9,044
		Alphabets	1,08,072	1,422	1,08,072
		Digits+Alphabets	1,17,116	1,541	
	Bad	Digits	4,446	59	~~4,446~~
		Alphabets	68,495	921	68,495
		Digits+Alphabets	72,941	960	72,941

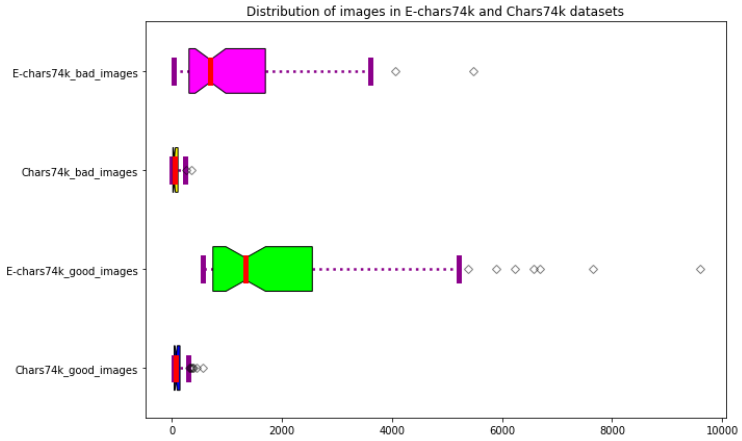


Fig. 5. Distribution of digits and alphabets images for the good and bad sets of the original Chars74k and the E-Chars74k datasets.

5. CNN Network Configuration

At first, all input images are normalized to 64x64 pixels size and converted into a grayscale image with 'zero padding'. Then, we applied a deep learning model based on CNN architecture to recognize digits and alphabets. Experiments on the Chars74k dataset and different subsets of the E-Chars74k dataset were carried out with the same network model.

5.1 Overview of CNN Architecture

Convolutional layer. The convolutional layer is the main key component of CNN architecture [48,50]. In this layer, feature maps have been generated from input images by applying learnable filters. Let $Image_{x,y}$ be an input image and $Kernel_{p,q}$ be the filter kernel. Then, the feature map may be represented as $Feature_{m,n} = Image_{x,y} * Kernel_{p,q}$. The actual size of feature maps depends on three parameters, i.e., number of kernels, stride size, and padding. Fig. 6 demonstrates the generation of a feature map from a sample input image and kernel.

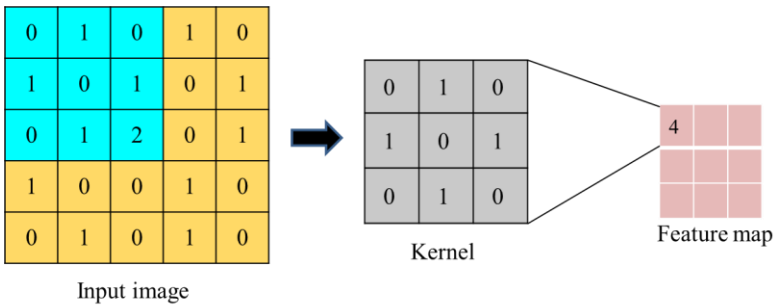


Fig. 6. Graphical representation of feature map generation from an input image and kernel.

In this layer, the non-linearity of a CNN model has been increased without affecting the previously generated feature map. Different types of activation functions like binary step, linear, sigmoid, tanh, ReLU, leaky ReLU, and parameterized ReLU are used in CNN. The most popular activation function is the Rectified Linear Unit (ReLU). The function of ReLU is expressed in (1).

$$f(y) = \begin{cases} 0, & \text{if } y < 0 \\ y, & \text{if } y \geq 0 \end{cases} \tag{1}$$

Pooling layer. The pooling layer is used to reduce the actual dimension of the feature map by using two different types of pooling functions, i.e., max pooling and average pooling. In the case of max pooling, the actual feature map window is reduced by the maximum value of the pooling window. In average pooling, the feature map window is reduced by the average value of the pooling window. Sub-sampling with max pooling is shown in Fig. 7.

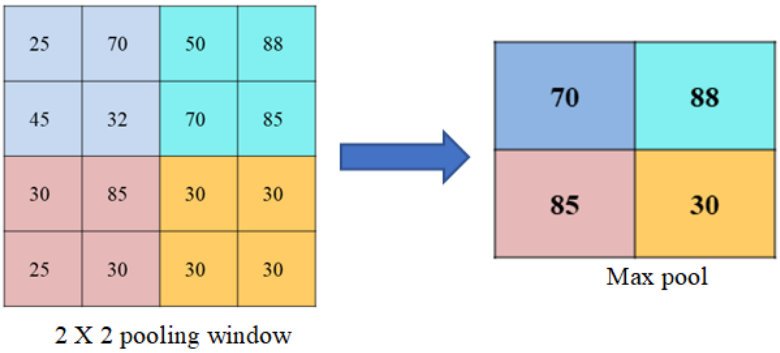


Fig. 7. Graphical representation of max pooling

Fully connected layer. In a fully connected layer, each neuron of the next layer is connected to the previous layer neurons. In the output layer, the softmax activation function predicts class probabilities.

Loss function. The model's performance is measured by the loss function computed based on the difference between expected and predicted class probabilities. Different loss functions like binary cross-entropy, mean absolute error, mean squared error, L1 loss, L2 loss, etc., are applied. The performance of a CNN model increases when the value of the loss function is gradually minimized.

Regularization. Over-fitting during training of CNN models is usually reduced by different regularization processes such as dropout, early stopping, L1 and L2 regularization, etc.

5.2 Applied CNN Pipeline

In our experiments, the 'Adam' optimizer is used in the CNN architecture shown in Fig. 8. In the first convolutional layer, 32 learnable convolutional filters with a kernel size of 5x5 pixels are convolved over an input image to generate the feature maps. Every single filter produces a separate feature map in two dimensions. We have applied 'zero padding' so that the size of feature maps remains the same as that of the input. After that, outputs are passed to the next layer through the ReLU activation function. Next, pooling layers with 2x2 sliding windows are added for higher-level feature extraction and computational cost reduction. After max-pooling, the size of feature maps gets reduced by 50%.

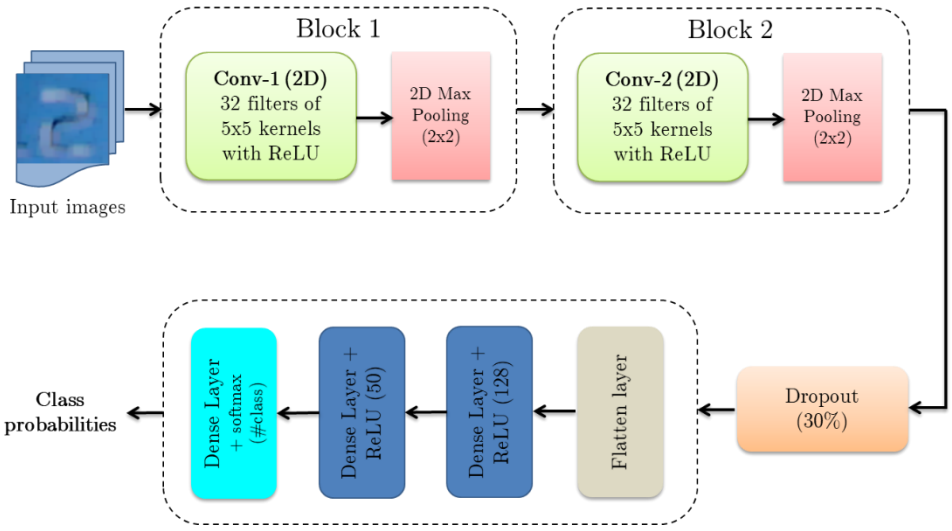


Fig. 8. The CNN model (shown as a block diagram) is applied in all the scene character recognition experiments conducted in this work.

In the second convolutional layer, 32 learnable convolutional filters with a kernel size of 5x5 pixels are applied for convolution with the ReLU activation function followed by 2x2 max-pooling. A dropout of 30% is applied to overcome the overfitting problem after the second convolutional layer. Fully connected layers contain 128 neurons each. Outputs of the neurons of the last layer are finally passed through a softmax function to obtain class probabilities. The highest among them corresponds to the predicted class.

6. Experiments and Results

This section presents each experiment's experimental setup and results with analysis and insights. The CNN model configuration with software and hardware requirements and the experimental setup are described first. Then, experimental results are presented, and a comparative view of the state-of-the-art methods and the proposed work is drawn. It may be noted that experiments involving training and prediction using the stated CNN model have been conducted on various subsets and the whole set of the augmented E-Chars74k dataset as well as the original Chars74k dataset.

Table 2. List of multiple parameters of the applied CNN model

Parameter	Value
Optimizer	Adam
Learning rate	0.001
Loss-function	Categorical cross-entropy
Padding	'Same'
Number of epoch	20 (experimentally chosen)
Batch size	500(experimentally chosen)

The proposed CNN model is trained using a batch size 500 with 20 epochs. The hyper-parameters are summarized in Table 2. The total number of trainable parameters is 840,110. A layer-wise summary of the CNN model, along with the number of trainable parameters generated by the model, is shown in Fig. 9. All experiments are conducted on a system with the hardware configuration of Intel(R) Core (TM) i5 CPU @ 3.20GHz, 8 GB RAM and 64-bit Windows 10

operating system. The software requirements include 'Python framework' (version 3.7) with 'Keras' libraries. Python programs are written in Spyder (Anaconda 3) notebook.

```
CNN MODEL SUMMARY:

Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d (Conv2D)              (None, 64, 64, 32)        2432
_________________________________________________________________
activation (Activation)      (None, 64, 64, 32)        0
_________________________________________________________________
max_pooling2d (MaxPooling2D) (None, 32, 32, 32)        0
_________________________________________________________________
conv2d_1 (Conv2D)            (None, 28, 28, 32)        25632
_________________________________________________________________
activation_1 (Activation)    (None, 28, 28, 32)        0
_________________________________________________________________
max_pooling2d_1 (MaxPooling2 (None, 14, 14, 32)        0
_________________________________________________________________
dropout (Dropout)            (None, 14, 14, 32)        0
_________________________________________________________________
flatten (Flatten)            (None, 6272)              0
_________________________________________________________________
dense (Dense)                (None, 128)               802944
_________________________________________________________________
activation_2 (Activation)    (None, 128)               0
_________________________________________________________________
dense_1 (Dense)              (None, 50)                6450
_________________________________________________________________
activation_3 (Activation)    (None, 50)                0
_________________________________________________________________
dense_2 (Dense)              (None, 10)                510
_________________________________________________________________
activation_4 (Activation)    (None, 10)                0
=================================================================
Total params: 837,968
Trainable params: 837,968
Non-trainable params: 0
_________________________________________________________________
```

Fig. 9. Layer-wise configurations of the applied CNN model (shape of data at each layer and the number of trainable parameters may be noted).

In this work, scene images of the Chars74k dataset (original good images and original bad images) and augmented E-Chars74k dataset (augmented good images and augmented bad images) are used in three groups of assays, i) only digits (0-9) recognition, ii) only alphabets (A-Z and a-z) recognition, and iii) digits and alphabets together. Results and findings of these assays are reported in Section 6.1-6.3.

6.1 Digits Recognition Result

In the Chars74k dataset, the total number of good images (digits only) is 595, and 292 bad images. These images are split into 80% for training and 20% for testing. Thus, the number of training data is 476, and testing data is 119 for good images (digits only), whereas training data is 234 and test data is 59 for bad images (digits only). As augmentations are applied to the training data of the original Chars74k images, the total number of training data for good images (digits only) in the E-Chars74k dataset is 9,044 and 4,446 for bad images (digits only). Testing data remain the same as the original dataset. Table 3 summarizes the prediction performance on all these subsets. It may be noted that augmentation has significantly improved the prediction performance compared to the original dataset. Confusion matrices of testing results are shown in Fig.10 and graphical representations of training and validation accuracy concerning epoch are shown in Fig. 11.

Table 3. Digits recognition accuracy of training, validation, and test samples of the original and augmented (good and bad) scene images.

Dataset	Total no. of sample	No. of training sample (80%)	Training Accuracy	Validation Accuracy	No. of test Sample (20%)	Test Accuracy	Precision	Recall	F1-score
Original good	595	476	58.59	45.36	119	42.98	50.47	37.46	43.00
Augmented good	9,044	9,044	97.65	93.86	119	73.77	74.66	71.17	72.87
Original bad	292	234	46.84	41.67	59	36.36	32.66	23.47	27.31
Augmented bad	4,446	4,446	95.94	87.76	59	53.61	42.52	42.86	42.86

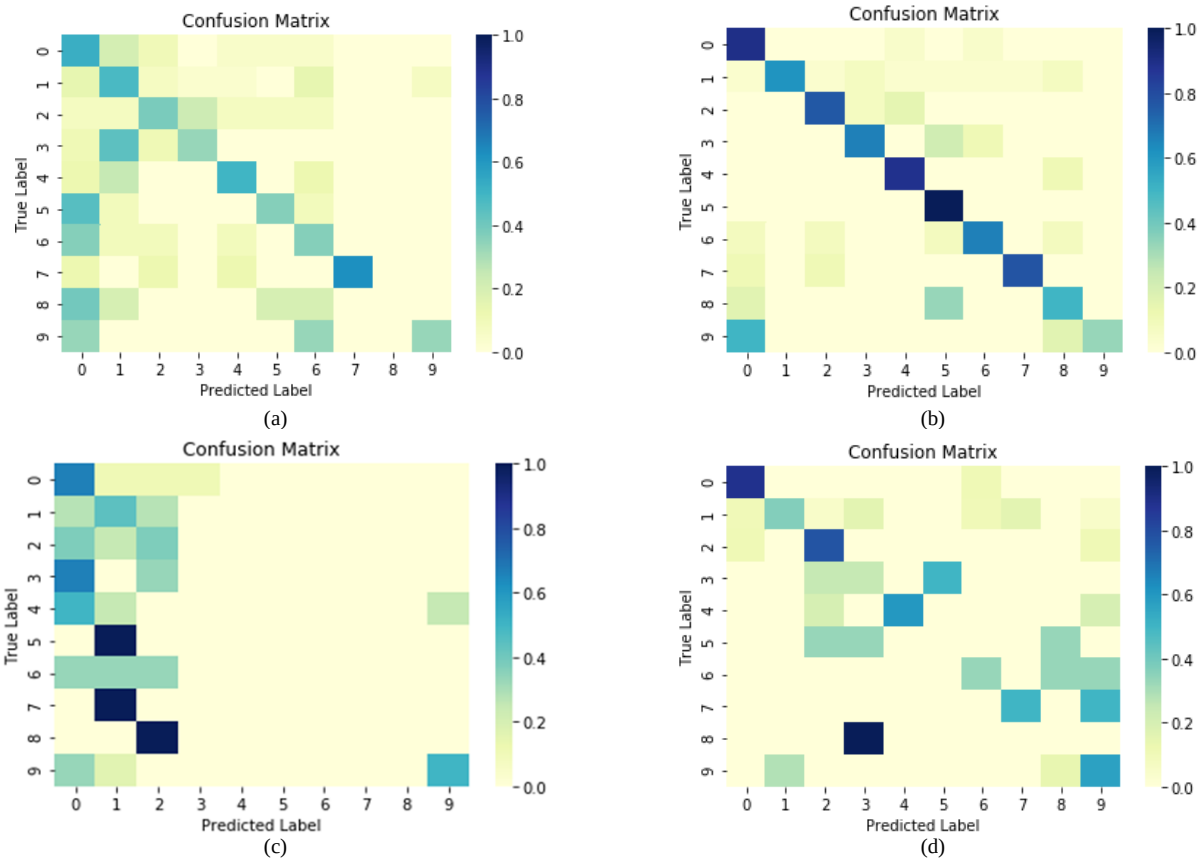


Fig. 10. Confusion matrices of the test samples for digits recognition (a) original good image, (b) augmented good images, (c) original bad image, (d) augmented bad images.

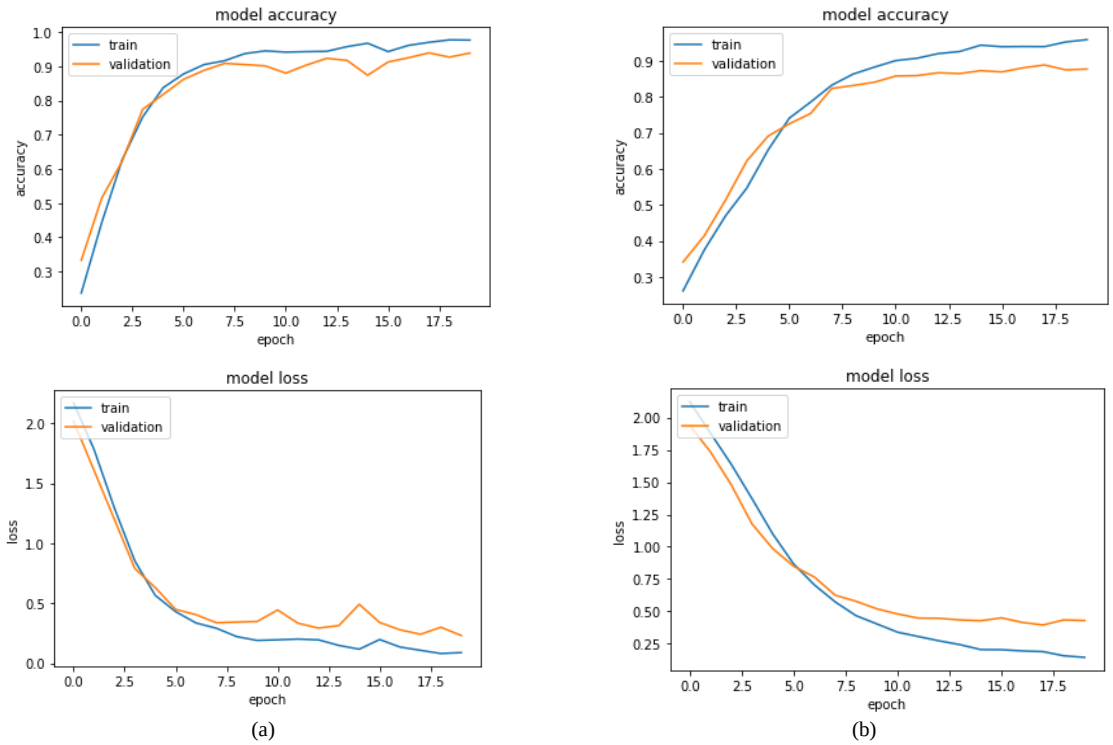


Fig. 11. Graphical representation for digits recognition of training vs. validation accuracy and loss concerning number of epoch during training time for (a) augmented good images, and (b) augmented terrible images.

6.2 Alphabets Recognition Result

In the Chars74k dataset, the total number of good images (Alphabets only) is 7,110, and there are 4,506 bad images (Alphabets only). The total numbers of images are split into 80% for training and 20% for testing. So, the number of training data is 5,688 and testing is 1,422 for good images (Alphabets only), whereas Training data is 3,605 and test data is 902 for bad images (Alphabets only). After 18 times, augmentations are applied to training data with the original training data of Chars74k images. The total number of training data for good images (Alphabets only) in the E-Chars74k dataset was 1,08,072 and 68,495 for bad images (Alphabets only). Testing data remain the same as the original dataset. After training the CNN model with epoch 20, the recognition result increased compared to the original dataset. Overall results are shown in Table 4, the confusion matrix of testing results is shown in Fig. 12, and graphical representations of training and validation accuracy concerning epoch are shown in Fig. 13.

Table 4. Alphabets recognition performance of training, validation, and test samples of augmented (good and bad) scene images of original Chars74k and augmented E-Chars74k dataset.

Dataset	Total no. of sample	No. of training sample (80%)	Training Accuracy	Validation Accuracy	No. of test Sample (20%)	Test Accuracy	Precision	Recall	F1-score
Original good	7,110	5,688	81.42	73.02	1,422	46.95	19.88	24.18	21.82
Augmented good	1,08,072	1,08,072	97.39	90.42	1,422	78.72	59.86	54.83	57.23
Original bad	4,506	3,605	45.48	36.26	921	33.80	19.13	18.67	18.89
Augmented bad	68,495	68,495	92.50	86.46	921	59.95	46.27	45.53	45.90

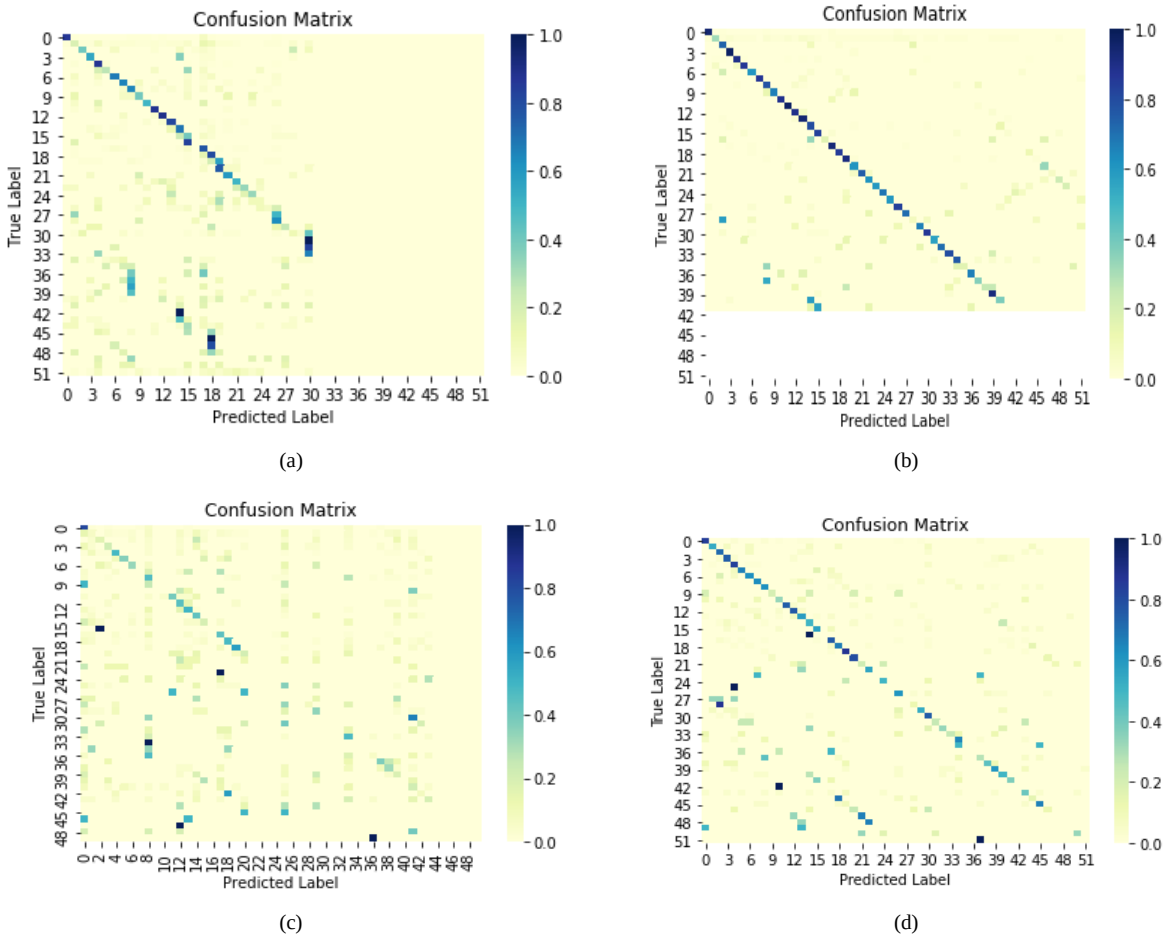


Fig. 12. Confusion matrices of the test samples for alphabets recognition, (a) original good images, (b) augmented good images, (c) original bad images, and (d) augmented bad images.

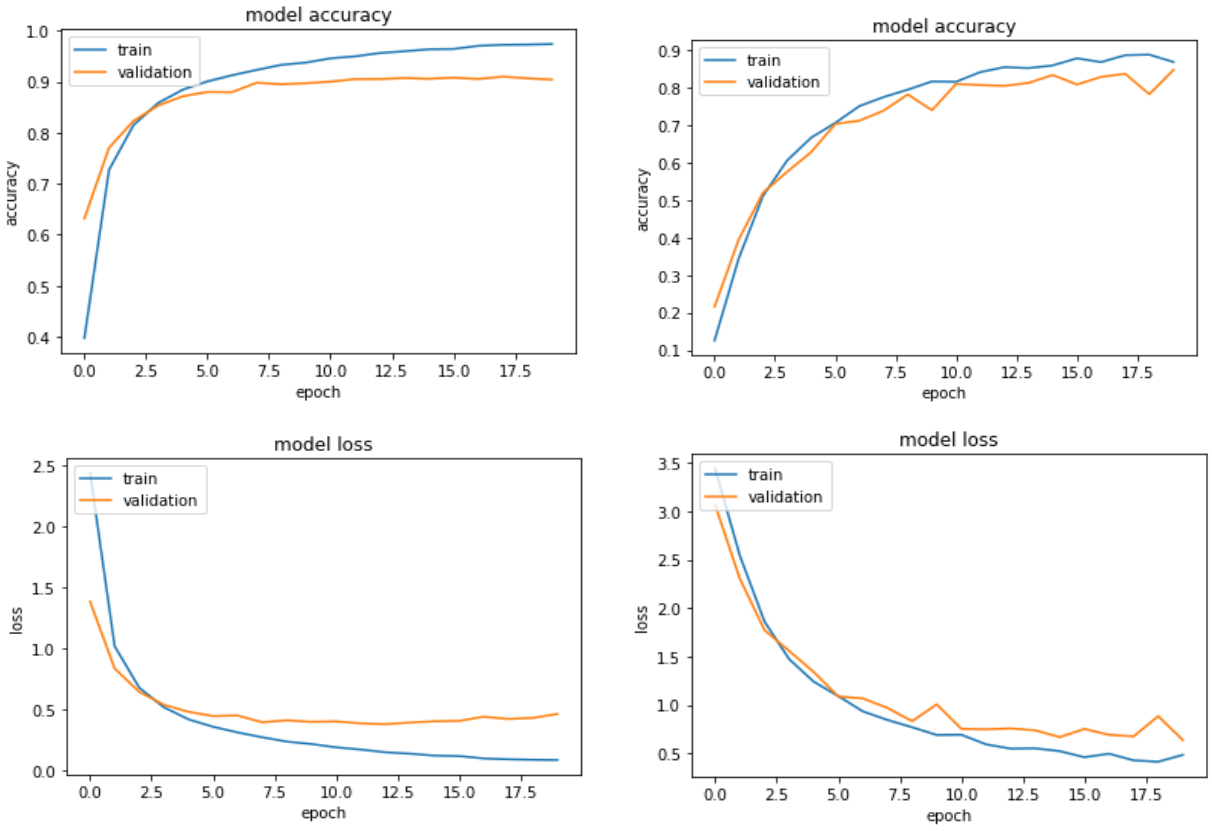


Fig. 13. Graphical representation for alphabet recognition of training vs. validation accuracy and loss with respect to several epochs during training time for (a) augmented good images and (b) augmented bad images.

6.3 Combined Digits and Alphabets Recognition

In the Chars74k dataset, the total number of good images (digits and alphabet only) is 7,705, and 4,798 bad images (digits and alphabet only). The total number of images is split into 80% for training and 20% for testing. So, the number of training data is 6,164, and testing is 1,541 for good images (digits and alphabets only). In contrast, Training data is 3,839 and test data is 960 for bad images (digits and alphabets only). After18 time augmentations were applied to training data with the original training data of Chars74k images, the total number of training data for good images (digits and alphabets only) in the E-Chars74k dataset was 1,17,116 and 72,941 for bad images (digits and alphabets only). Testing data remain the same as the original dataset. After training the CNN model with epoch 20, the recognition result increased compared to the original dataset. Overall results are shown in Table 5, the confusion matrix of the testing results is shown in Fig. 14, and graphical representations of training and validation accuracy concerning epoch are shown in Fig. 15.

Table 5. Training, validation, and prediction performance of digits and alphabets together for the original and augmented dataset.

Dataset	Total no. of sample	No. of Training sample (80%)	Training Accuracy	Validation Accuracy	No. of Test sample (20%)	Test Accuracy	Precision	Recall	F1-score
Original good	7,705	6,164	56.75	50.41	1,541	59.95	43.37	42.53	42.94
Augmented good	1,17,116	1,17,116	97.47	90.32	1,541	70.99	62.02	60.40	61.19
Original bad	4,798	3,839	30.94	23.33	960	25.91	11.84	11.33	11.57
Augmented bad	72,941	72,941	94.83	85.31	960	51.07	35.84	36.43	36.13

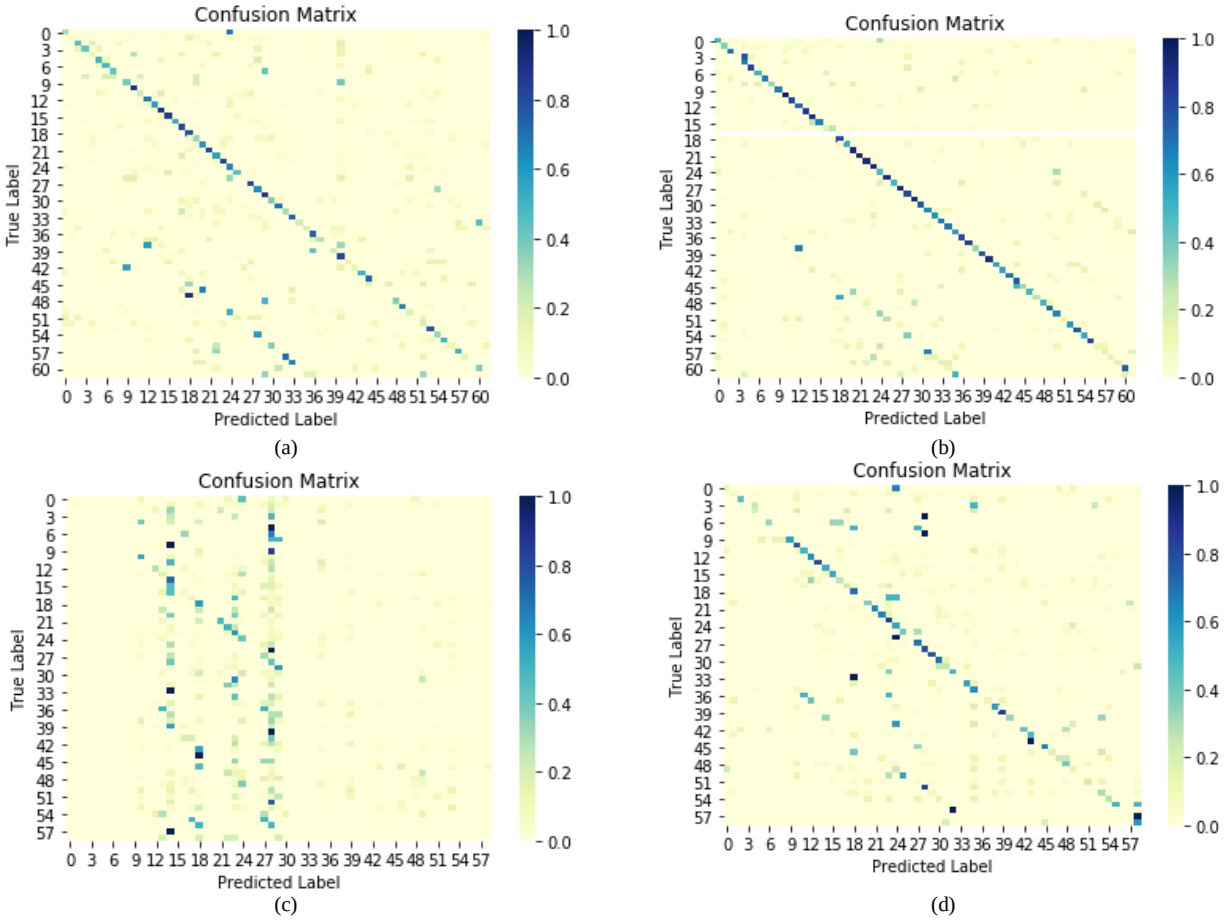


Fig. 14. Confusion matrices of test sample for digits and alphabets recognition, (a) original good image, (b) augmented good image, (c) original bad images, and (d) augmented bad images.

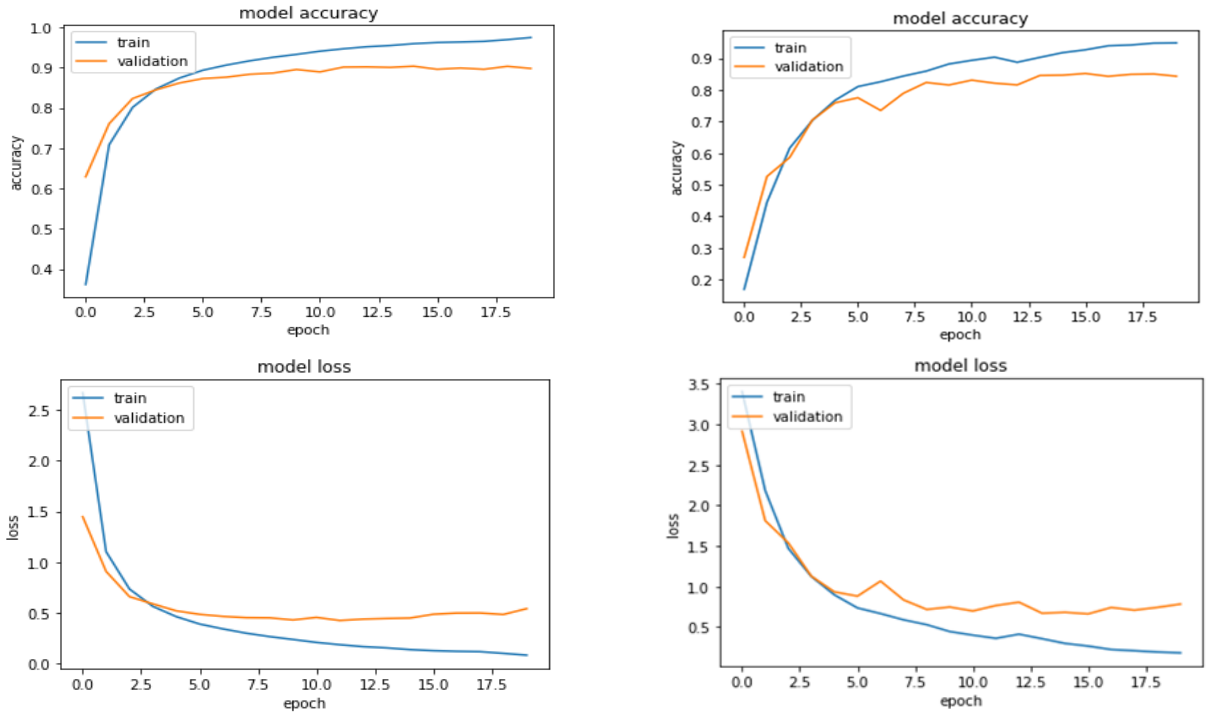


Fig. 15. Graphical representation for digits and alphabets together recognition of training vs. validation accuracy and loss with respect to the number of epochs during training time for (a) augmented good images and (b) augmented bad images

6.4 Discussion

The Chars74k dataset has been experimented with different deep learning and traditional character recognition methods. Authors enlarged Chars74k datasets in their own way for training deep learning models. A comparative analysis of the recognition accuracy of Chars74k and E-Chars74k is described in Table 6.

Table 6. Performance analysis before and after augmentation on Chars74k datasets and E-Chars74k datasets

Dataset/subset	Chars74k Accuracy	E-Chars74k Accuracy
Good (Digits)	42.98	73.77
Good (Alphabets)	46.95	78.72
Good (Digits+Alphabets)	59.95	70.99
Bad (Digits)	36.36	53.61
Bad (Alphabets)	25.91	59.95
Bad (Digits+Alphabets)	33.80	51.07

Although, some other experiments give extraordinary performance parameters, they vary from experiment to experiment. Abdali et al. [26] trained the proposed model with 92% accuracy using the EMNIST dataset combined with the Chars74k dataset with random augmentation—Chandio et al. [24] trained with the Chars74k dataset (bad and good images). The total number of images is 12,402 only, and without augmentation, the proposed method has been trained by CNN. The experimental accuracy of this method is 81.45%. Another experiment was done by Arroyo-Perez et al. [23] with 90% accuracy using the CNN model and trained by the Chars74k dataset along with 33,849 images. In this paper, the experimental observations are dissimilar to the research mentioned above works. Chars74k scene images are divided into training (80%) and test (20%) data. Only train images are augmented 18 times and construct E-Chars74k. The number of images in E-Chars74k becomes 1,92,558 with 80% of the original training data increased 18 times and 20% of the actual test data. Experiments are done in three ways i.e.: digit, alphabet, and digits with alphabets.

7. Conclusion

In this work, we propose a CNN model-based recognition method for scene character datasets. Simultaneously, a large dataset called E-Chars74k (extended Chars74k) is prepared for scene character recognition experiments. There are two scene character datasets viz. Chars74k and E-Chars74k that are trained using a CNN model. Both datasets contain digits and alphabets. Due to the diverse augmentation with machine learning insights, E-Chars74k produces reasonably satisfactpry results. The benchmark performance is also presented for E-Chars74k. Obtained accuracies on the said dataset for good images are 73.77% (digits), 78.72% (alphabets), and 70.99% (digits and alphabets together), and for highly intricate images, 53.61% (digits), 59.95% (alphabets) and 51.07% (alphabets and digits together).

The dataset along with annotations and benchmarking framework is now open to the research fraternity for evaluating their methods, designing superior algorithms, and thereby fostering the development of robust scene text understanding methods. In future, more efficient deep-learning network may be constructed for scene character recognition and the same may be applied for end-to-end scene word or image recognition. Impact of different augmentation techniques on the network training and prediction performance may also be investigated.

Acknowledgements

The first author expresses sincere gratitude to the Department of Computer Science and Engineering, Brainware University, Kolkata, India, for their support in conducting this research. T. Khan extends thanks to the School of Computer Science and Engineering, VIT-AP University, for providing the necessary support to carry out this work.

References

[1] T. Khan, R. Sarkar and A. F. Mollah, “Deep learning approaches to scene text detection: a comprehensive review,” Artificial Intelligence Review, vol. 54, no. 5, pp. 3239-3298, Springer, 2021.
[2] H. Lin, P. Yang and F. Zhang, “Review of Scene Text Detection and Recognition,” Archives of Computational Methods in Engineering, vol. 27, no. 2, pp. 433-454, Springer, 2019.
[3] P. Sengupta and A. F. Mollah, "Journey of scene text components recognition: Progress and open issues," Multimedia Tools and Applications, vol. 80, no.4, pp. 6079-6104, Springer, 2020.
[4] S. Saha, N. Chakraborty, S. Kundu, S. Paul, A. F. Mollah, S. Basu and R. Sarkar, “Multi-lingual scene text detection and language identification,” Pattern Recognition Letters, vol. 138, pp. 16-22, Elsevier, 2020.
[5] W. Liu, C. Chaofeng and K. Wong, "SAFE: Scale Aware Feature Encoder for Scene Text Recognition," In Proceedings of Asian Conference on Computer Vision, pp. 196-211, Springer, 2018.
[6] C. Kang, G. Kim and S. Yoo, “Detection and recognition of text embedded in online images via neural context models,” In Proceeding of Association for the Advancement of Artificial Intelligence, pp. 4103–4110, 2017.
[7] F. Zhan and S. Lu, “Esir: End-to-end scene text recognition via iterative image rectification,” In Proceedings of the Conference on Computer Vision and Pattern Recognition, pp. 2059-2068, IEEE, 2019.

[8] B. Shi, X. Bai and C. Yao, “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, vol. 39, no. 11, pp. 2298-2304, IEEE Transactions, 2016.
[9] H. Liu and B. Bir, "Pose-Guided R-CNN for Jersey Number Recognition in Sports," In Proceedings of Conference on Computer Vision and Pattern Recognition Workshops, pp. 7297-7306, IEEE, 2019.
[10] D. Karatzas, F. Shafait, S. Uchida and M. Iwamura, “ICDAR 2013 robust reading competition”, In Proceedings of 12th International Conference on Document Analysis and Recognition, pp. 1484-1493, IEEE, 2013.
[11] M. Iwamura, “Advances of Scene Text Datasets”, arXiv preprint arXiv:1812.05219, 2018.
[12] N. Nayef, Y. Patel, M. Busta, P. N. Chowdhury, D. Karatzas, W. Khlif, J. Matas, U. Pal, J. C. Burie, C. L. Liu and J. M. Ogier, “ICDAR2019 Robust Reading Challenge on Multi-lingual Scene Text Detection and Recognition--RRC-MLT-2019”, arXiv preprint arXiv:1907.00945, 2019.
[13] D. Karatzas, L. Gomez-Bigorda, A. Nicolaou, S. Ghosh, A. Bagdanov, M. Iwamura, J. Matas, L. Neumann, V. R. Chandrasekhar, S. Lu, and F. Shafait, “ICDAR 2015 competition on robust reading,” In Proceeding of 13th International Conference on Document Analysis and Recognition (ICDAR), pp. 1156-1160, IEEE, 2015.
[14] S. M. Lucas, A. Panaretos, L. Sosa, A. Tang, S. Wong and R. Young, “ICDAR 2003 robust reading competitions”, In Proceedings of Seventh International Conference on Document Analysis and Recognition, pp. 682-687, IEEE, 2003.
[15] T. E. De Campos, B. R. Babu and M. Varma, "Character recognition in natural images," In Proceedings of International Conference on Computer Vision Theory and Application, vol. 2, no. 7, pp. 273-280, 2009.
[16] S. Long, X. He and C. Ya, “Scene Text Detection and Recognition: The Deep Learning Era,” International Journal of Computer Vision, vol. 129, no. 1, pp. 161-184, Springer, 2021.
[17] E. A. Enriquez, N. Gordillo, L.M Bergasa, E. Romera and C.G. Huélamo, “Convolutional neural network vs. traditional methods for offline recognition of handwritten digits,” In Workshop of Physical Agents, pp. 87-99, Springer, 2018.
[18] P. Sengupta and A.F. Mollah, “Dissected Scene Character Recognition using HOG Descriptors,” In Internet of Things and Its Applications, pp. 199-209, Springer, 2021.
[19] B. Chekol, N. Celebi and T. Tasci, “Segmented character recognition using curvature-based global image feature,” Turkish Journal of Electrical Engineering & Computer Sciences, vol. 27, no. 5, pp. 3804-3814, 2019.
[20] C. Y. Lee, A. Bhardwaj, W. Di, V. Jagadeesh and R. Piramuthu, “Region-based discriminative feature pooling for scene text recognition,” In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 4050-4057, 2014.
[21] P. Sengupta and A. F. Mollah, “Scene Character Recognition with Morphological Filtering and HOG Features,” In Soft Computing Techniques and Applications, Springer Nature AISC, vol. 1248, pp. 1-9, Springer Nature, 2020.
[22] M. Ali and H. Foroosh, “A holistic method to recognize characters in natural scenes,” In Proceedings of International Conference on Computer Vision Theory and Applications, pp. 449-457, 2016.
[23] L. Neumann and J. Matas, “A method for text localization and recognition in real-world images,” In Proceedings of Asian Conference on Computer Vision, pp. 770-783, Springer, 2021.
[24] A. Coates, B. Carpenter, C. Case, S. Satheesh, B. Suresh, T. Wang, D. J. Wuand, A. Y. Ng, “Text detection and character recognition in scene images with unsupervised feature learning,” In Proceedings of International Conference on Document Analysis and Recognition, pp. 440-445, IEEE, 2011.
[25] D. E. Arroyo-Pérez, O. I. Álvarez-Canchila, A. Patiño-Saucedo, H. R. González, A. Patiño-Vanegas, “Automatic recognition of Colombian car license plates using convolutional neural networks and Chars74k database”, Journal of Physics, vol. 1547, no. 1, pp. 12-24, 2020.
[26] A. A. Chandio, M. Asikuzzaman, and M. R. Pickering, “Cursive Character Recognition in Natural Scene Images Using a Multilevel Convolutional Neural Network Fusion”, IEEE Access, vol. 8, pp. 109054–109070, 2020.
[27] J. H. Lin, J. Lazarow, A. Yang, D. Hong, R. Gupta, and Z. Tu, “Local binary pattern networks,” In Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision, pp. 825-834, IEEE, 2020.
[28] A. R. Abdali, R. F. Ghani, “Robust character recognition for optical and natural images using deep learning,” In Proceedings of Student Conference on Research and Development (SCOReD), pp. 152-156, IEEE, 2019.
[29] Z. Zhang, H. Wang, S. Liu, T. S. Durrani, “Bilateral convolutional activations encoded with Fisher vectors for scene character recognition,” IEICE Transactions on Information and Systems, vol. 10, no. 5, pp. 1453-1459, 2018.
[30] S. B. Driss, M. Soua, R. Kachouri, and M. Akil, “A comparison study between MLP and convolutional neural network models for character recognition,” In Proceedings of Real-Time Image and Video Processing, pp. 10223-10229, 2017.
[31] Z. Zhang, H. Wang, S. Liu, and B. Xiao, “Consecutive convolutional activations for scene character recognition,” IEEE Access, vol. 28, no. 3, pp. 35734-35742, 2018.
[32] D.N. How and K. S. Sahari, “Character recognition of Malaysian vehicle license plate with deep convolutional neural networks,” In Proceedings of IEEE International Symposium on Robotics and Intelligent Sensors (IRIS), pp. 1-5, 2016.
[33] A. Ray, S. Rajeswar, and S. Chaudhury, “Scene text analysis using deep belief networks,” In Proceedings of the Indian Conference on Computer Vision Graphics and Image Processing, pp. 1-8, 2014.
[34] Y. Zhang, W. Wang, and L. Wang, “Scene text recognition with deeper convolutional neural networks,” In Proceedings of IEEE International Conference on Image Processing (ICIP), pp. 2384-2388, IEEE, 2015.
[35] M. Ali, and H. Foroosh, “Natural Scene Character Recognition without Dependency on Specific Features,” In Proceedings of Computer Vision Theory and Applications, pp. 368-376, 2015.
[36] Y. Li, S. Zhang, X. Zhou, and F. Ren, “Build a compact binary neural network through bit-level sensitivity and data pruning,” Neurocomputing, pp. 45-54, Elsevier, 2020.
[37] Y. Wang, C. Shi, B. Xiao, and C. Wang, “Learning spatially embedded discriminative part detectors for scene character recognition,” In Proceedings of 14th IAPR International Conference on Document Analysis and Recognition (ICDAR), pp. 363-368, IEEE, 2017.
[38] A. Hernández-García, and P. König, “Further advantages of data augmentation on convolutional neural networks,” In Proceedings of International Conference on Artificial Neural Networks, pp. 95-103, Springer, 2018.
[39] E. D. Cubuk, B. Zoph, D. Mane, V. Vasudevan, Q.V. Le, “Autoaugment: Learning augmentation policies from data,” arXiv preprint arXiv: 1805.09501, 2018.

[40] C. Shorten and T.M. Khoshgoftaar, “A survey on image data augmentation for deep learning,” Journal of Big Data, vol. 6, no. 1, pp. 1-48, Springer, 2019.
[41] J. M. B. Francisco, S. Fiammetta, M. J. Jose, U. Daniel, F. Leonardo, “Forward noise adjustment scheme for data augmentation,” arXiv preprints. 2018.
[42] D. Dua and T. E. Karra, “UCI machine learning repository,” Irvine, CA: The University of California, School of Information and Computer Science, 2017.
[43] K. Guoliang, D. Xuanyi, Z. Liang and Y. Yi, “PatchShuffle regularization,” arXiv preprint. 2017.
[44] H. Kaiming, Z. Xiangyu, R. Shaoqing, and S. Jian, “Deep residual learning for image recognition,” In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 770-778, IEEE, 2016.
[45] I. Hiroshi, “Data augmentation by pairing samples for images classification,” arXiv:1801.02929v2, 2018.
[46] V. Terrance and W. T. Graham, “Improved regularization of convolutional neural networks with cutout,” arXiv:1708.04552v2, 2017.
[47] A. Hernández-García and P. König, “Further advantages of data augmentation on convolutional neural networks,” In Proceedings of International Conference on Artificial Neural Networks, pp. 95-103, Springer, 2018.
[48] T. Khan and A. F. Mollah, “Component-level Script Classification Benchmark with CNN on AUTNT Dataset,” In Proceedings of International Conference on Frontiers in Computing and Systems, pp. 225-234, Springer, 2021.
[49] T. Khan and A. F. Mollah, “AUTNT-A component level dataset for text non-text classification and benchmarking with novel script invariant feature descriptors and D-CNN,” Multimedia Tools and Applications, vol. 78, no. 22, pp. 32159-32186, Springer, 2019.
[50] P. Ahamed, S. Kundu, T. Khan, V. Bhateja, R. Sarkar and A. F. Mollah, “Handwritten Arabic numerals recognition using convolutional neural network,” Journal of Ambient Intelligence and Humanized Computing, vol. 11, no. 11, pp. 5445-5457, Springer, 2020.
[51] J. H. Lin, J. Lazarow, A. Yang, D. Hong, R. Gupta, and Z. Tu, "Local binary pattern networks." In Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision, pp. 825-834. IEEE, 2020.

Authors’ Profiles

Payel Sengupta was awarded Ph.D. in Computer Science and Engineering from Aliah University, West Bengal in 2025. She is currently working as an Assistant Professor in the Department of Computer Science and Engineering at Brainware University, Kolkata, West Bengal. She received her Master of Technology (M. Tech) degree in Computer Science and Engineering from Maulana Abul Kalam Azad University of Technology (MAKAUT, formerly WBUT) in 2013. Her research interests include Machine Learning, Digital Image Processing, and Deep Learning. She has published several articles in leading journals and conferences, contributing to advancements in scene image analysis. Her research has been recognized for its innovative approach and has garnered attention within the academic community.

Tauseef Khan has received the Ph.D. degree in Computer Science and Engineering from Aliah University, Kolkata in 2022. He completed Master of Technology (Gold) in Information Technology from Maulana Abul Kalam Azad University of Technology (MAKAUT, formerly WBUT) in 2013. He is currently working as an Assistant Professor in the School of Computer Science and Engineering, VIT-AP University, Amaravati, Andhra Pradesh. His research interest includes computer vision, digital image processing, machine learning, pattern recognition, etc. He has published several articles in reputed journals and conferences on scene text detection, image classification, script identification, etc. He is a member of IEEE and life member of ISTE.

Ayatullah Faruk Mollah, PhD(Engg) is an Assistant Professor and former Head(Officiating) of the Department of Computer Science and Engineering, Aliah University, India. He has completed his doctoral study from Jadavpur University, India. He is a senior member of IEEE and IU-IAPR. He was a Senior Software Engineer at Atrenta (I) Pvt. Ltd. Noida, India. He has also worked with ScanBiz Mobile Solutions, New York, USA. He was awarded prestigious European Union Erusmus Mundus cLink Fellowship, University Grants Commission Research Fellowship, and iCell postdoctoral fellowship at University of Warsaw, Poland. His research interests include deep learning, data science, image and video analysis, machine learning, bioinformatics, etc. So far he has authored nearly hundred articles in refereed journals and conference proceedings. Dr. Mollah is also a Co-Principal Investigator of a project funded by DST, Govt. of India.

How to cite this paper: Payel Sengupta, Tauseef Khan, Ayatullah Faruk Mollah, "E-Chars74k: An Extended Scene Character Dataset with Augmentation Insights and Benchmarks", International Journal of Image, Graphics and Signal Processing(IJIGSP), Vol.17, No.6, pp. 133-150, 2025. DOI:10.5815/ijigsp.2025.06.08