

Bangla Handwritten Character Recognition using Deep Convolutional Neural Network for multi-label image annotation

Sabbir Ahmed · Manan Binth Taj Noor · Mohammad Abu Yousuf

Received: November 16, 2022/ Accepted: date

Abstract

Recognition of handwritten compound characters still poses a bigger challenge in Bangla OCR system owing to structural complex shapes comprising a higher rate of resemblance among the characters and a large number of classes. Hence, a multi-labelling technique based on the conjoining relationship of the compound characters has been proposed in this paper. Data representation in the proposed technique depletes the classification complexity alleviating the obligation of disparate classes of compound characters restricting to only recognition of the consonant character's classes carving those particular compound characters. Then, the efficacy of the multi-label technique implemented on the compound characters of CMATERdb 3.1.3.2 dataset and consonant and vowel character of Bengali.AI dataset, Ekush dataset have been investigated by a Deep Convolutional Neural Network(DCNN) architecture in a supervised fashion for classification. Classification performance of the DCNN architecture has been augmented through exertion of thresholding and layer wise parameter tuning. Experimental results show that, the proposed multi-labelling technique equipped with DCNN architecture provides coherent classification performance from the perspectives of proving competence of the multi-labelling technique conforming recognition of different compound characters. Furthermore, ran-

domly collected complex shaped written compound characters have been tested as well on the proposed architecture in order to manifest the competence of the multi-labelling technique of this work.

Keywords Deep Learning · Bangla OCR · Bangla compound character · Multi-label Bangla OCR · CNN

1 Introduction

Optical character recognition (OCR) is a technique for offline computer recognition of linguistic characters from digital image formats created by scanning documents in either printed or handwritten form. OCR's objective is to extract textual data from scanned documents or other data formats [1,2]. General-purpose OCR from the digitized image mainly consists of scanning, locating and segmentation, preprocessing, feature extraction and finally classification in various methods [3]. OCR is a vast field with various applications such as handwriting recognition [4], invoice imaging [5], legal industry [5], banking, health care industry [5]. OCR is also widely used in many other aspects like optical music recognition without any human correction or human effort [6], captcha [7], institutional repositories with digital libraries [8] and automatic number plate recognition [9,10].

Handwritten and printed character recognition is a common research topic and a good number of works have also been published on different language such as English [11,12], Roman scripts, mandarin Chinese [13,14], Japanese [15], Korean, Arabic [16,17], Russian [18] etc. Whereas the scope of some influential and high-ranking Asian scripts like Oriya, Tamil, Devanagiri and Bangla have been explored very insignificantly in OCR research. Of all these, Bangla being the national

Sabbir Ahmed (✉)
IIT, Jahangirnagar University, Dhaka – 1342, Bangladesh
E-mail: sabbir.iit.ju@gmail.com

Manan Binth Taj Noor (✉)
IIT, Jahangirnagar University, Dhaka – 1342, Bangladesh
E-mail: manan.noor@juniv.edu

Mohammad Abu Yousuf (✉)
IIT, Jahangirnagar University, Dhaka – 1342, Bangladesh
E-mail: yousuf@juniv.edu

language of Bangladesh, the second most popular language of India holds a lot of importance. Following that Bangla is also the fifth most popular language of the world which validates the need to be prospected in the field of OCR. Moreover, most of these research efforts in Bangla OCR have been conducted with basic characters which comprise only one eighth (1/8) of the characters of Bangla script [19]. On the other hand, complex-shaped compound characters adding up to nearly eighty-five percent (85%) of the total number of Bangla characters have been hardly addressed [20, 21] in this backdrop. The written format of the Bangla language includes 11 vowels, 39 consonants, 10 modifiers and 280 compound characters [19] conveying a structurally complex scripting nature. Bangla compound characters consist of two or more consonants which most of the time are adjoined in a disrupt format with fewer characteristics resembling isolated versions of those consonants. Additionally, the difference between compound characters changes slightly in terms of curvature. Along with compound characters, the complete Bangla handwritten format also consists of consonants and vowels also. Owing to these structural complex representations and having more than 280 classes for compound characters and 50 classes for basic characters; Bangla handwritten character recognition is a formidable pattern recognition problem still now. Thus, this present work is motivated to address the problem of recognizing the complete Bengali handwritten character (compound character, vowel and consonant character) of the Bangla script.

Different machine learning methods have been applied in this context; such as Multilayer Perceptron, Radial Basis Function, Global and Local Feature Extraction, Geometric Context, statistical feature-based vector, histogram of oriented gradients being taken the bi-directional and multi-dimensional Long short-term memory. But the most prominent way tends to be the application of deep learning which proved to be unanimously very useful in image processing with a very high recognition rate [22]. Moreover, in Bangla OCR, printed character segmentation has been mostly conducted by fuzzy multifactorial analysis [23] whereas recognition has been mostly carried out by structural-feature-based tree classifier [24] and Digital Curvelet Transform [25]. On the other hand, handwritten character recognition is mostly conducted by SVM based hierarchical learning architectures [26], soft computing paradigm embedded in two pass approach [27]. But most recently deep learning has shown major improvements in handwritten Bangla OCR with satisfactory recognition and classification accu-

racy. Following that, this work addresses the application of deep CNN architecture to recognize both multi-label handwritten Bangla Compound characters and single-label basic characters by a single classifier. Figure 1 shows the usage of different approaches implemented till now in terms of the Bangla OCR task.

While multi-labeling in handwritten character recognition is uncommon, multi-labeling in image classification is well-established in the research community. Generally, a single image can contain multiple objects that must be classified. When objects are disjoint and do not overlap, the categorization problem can be solved using image segmentation [28]. Each segmented image contains just one object; thus, the single-label classification technique can be used. However, in the circumstance of overlapping objects, the position of each entity may result in a different classification. Multi-label may handle these relationships by utilizing several instances of the label. Due to the fact that Bangla compound letters contain several consonants, this issue can be reduced to the classification of multiple consonants with respect to their relative location [29]. Moreover, since the compound character recognition problem can be reduced to the recognition of basic characters, this methodology can also be applied to any basic character recognition within the same classifiers. Thus, a single multi-label classifier could be adequate to classify consonants and, vowels with compound characters.

Most of the approaches taken previously for single-label classification perspective [24, 23, 26, 27, 30, 31, 32, 33] are based on structural feature set to handle complex structural shapes. The majority of the works focused on the shape decomposition of the compound characters. And then, supervised layer-wise trained DCNN had been applied to classify each character into an individual class or a group despite the fact that compound characters consist of multiple consonants. Some other approaches directly classify all the consonants, vowels, and compound characters as a single category. Addressing these observations of character recognition in the premises of Bangla OCR, the following contributions are put forward in the current work:

- Multi-labelled dataset has been developed based on the the compound characters of CMATERdb 3.1.3.2, and bangla vowel consonant characters from AI bangla and matrivasha datasets. Ekush and CMATERdb basic dataset were single labeled but compatible of the multi-labeling. As far as the studies conducted for Bangla character data representation in datasets, this aspect of multi-labelling with compatible single labeling has not been attempted and explored till now.

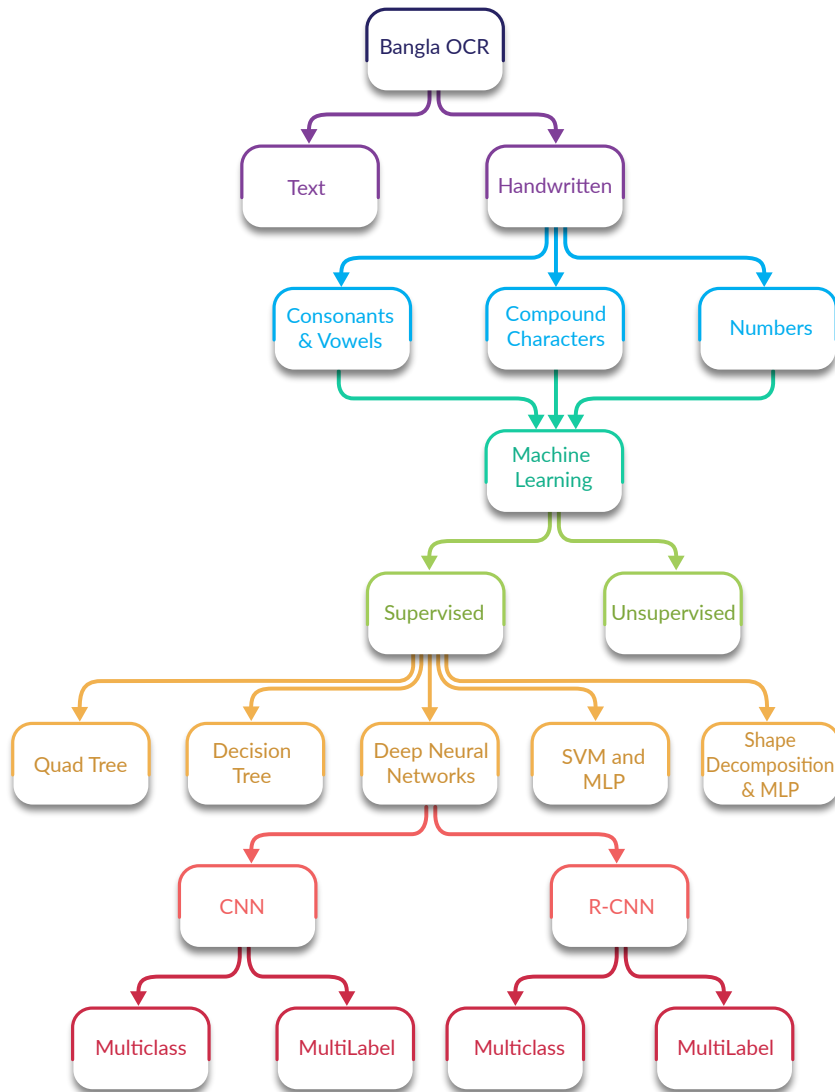


Figure 1: Possible approaches which can be applied for Bangla OCR

- Instead of classification and recognition of a complex-shaped conjunct character, classification and recognition of multiple basic characters constituting the compound character have been conducted through multi-labelling.
- Since the same set of consonants can form multiple compound characters, thus based on their relative position, relationship between those consonants have also been uniquely identified for each of the compound characters.
- Basic consonants and vowel are also labeled as a single characters for the multi-label paradigm.
- A fusion multi-label and single-label multi-class Deep Convolutional Neural Network architecture have been developed for complete Bangla handwritten character recognition. The classifier is proposed to classify

each consonant and their conjugating relationship in one compound character. It has ultimately resulted in a higher recognition rate in recognizing the compound character than recognition of any conjunct as a separate class.

- The proposed CNN architectures also tested for Consonants and vowels, which are also part of the multi-labeling scheme.

The rest of the article is outlined as: section 2 delineates the related works, section 3 describes the methodology undertaken for this work, section 4 includes experimental setup and evaluation measures; section 5 presents experimental results and discussion, and section 6 concludes the work with future perspectives.

2 Related Work

Prior research on multi-label single-label fusion classification in character recognition is scarce. Additionally, no attempt of fusion multilabel Bangla handwritten character identification has been conducted, and this work represents the first step in that direction, according to the research done for this work. So, learning about the Bangla compound and the fundamental character recognition framework for single-label categorization in both printed and handwritten versions was part of the research related with the proposed model. However, compared to the literature on basic and numerical character identification in Bangla, research on Bangla compound character recognition is rather scant.

Following that, a very early attempt at Bangla character recognition was discovered, which identified printed Bangla number, basic, and compound characters. A structural-feature-based tree classifier was used in that study, which was followed by document scanning, skew correction, text graphics separation, line segmentation, zone identification, word, and character segmentation. Finally, there was a character-level identification accuracy of 99.10 percent [24]. Another study employed a multilayer perceptron with Quad-tree structure, Longest Run, and Cross-Validation to detect the most commonly used 55 compound characters, with an average recognition rate of 84.67 percent after 3 fold cross-validation [23]. Then, for handwritten Bangla compound character recognition, Das et al. suggested an MLP and SVM-based framework [33]. Only the first 43 of 180 compound characters were evaluated for this study, based on their frequency of occurrence in current Bangla literature. Quad tree-based shadow features and longest run features were used in this study, with MLP and SVM classifiers achieving average recognition rates of 79.83 percent and 80.5107 percent, respectively. Then, for Bangla fundamental characters, modifiers, and compound characters [27], a soft computing paradigm based on a two-pass technique was presented. They developed an algorithm to determine hull-based and longest-run features with SVM classifiers where 87.26% recognition accuracy was obtained. A supervised deep convolutional neural networks architecture has been introduced in [34]. In this work, sequential multiple convolution-pool layers and fully connected layer were used with the RMSProp algorithm to achieve faster convergence with 90.33% recognition accuracy using the CMATERdb 3.1.3.2 dataset. Following that, a shape decomposition-based segmentation technique has also been proposed to decompose the compound characters into basic char-

acter shape components with 88.74% recognition accuracy [35].

In recent work, Ferdous et al. [36] have presented the Matrivasha dataset for the recognition of 120 compound characters in Bangla. The dataset consists of 306,240 pictures published by 2552 individual Bangladeshi authors. They have achieved an accuracy of 94.49% in the Matrivasha dataset. Another dataset generated by Hasan et al. [37] has 249,911 hand-written photographs of 171 compound Bangla characters and 80,403 hand-written pictures of 50 basic Bangla characters. The dataset was named AIBangla and written by over 2,000 individual authors from different institutions throughout Bangladesh. They attained a maximum accuracy of 98.13 percent for basic character classes and 81.83 percent for compound character classes on the AIBangla dataset's test set. Rabby et al. developed the Ekush dataset [38], which comprises Bangla numerical digits, vowels, consonants, modifiers, and compound characters. The database comprises 367,018 discrete hand-written characters authored by 3086 separate authors.

Khan et al [?] presents a deep CNN (Convolutional Neural Network) model using the SE-ResNeXt. The squeeze and excitation (SE) blocks are added with existing ResNeXt to address the Bangla handwritten compound character recognition. To validate the performance of the proposed model, they have utilized Mendeley BanglaLekha-Isolated 2 dataset. The experimental results demonstrate that the proposed model exhibits an average accuracy of 99.82% in recognizing Bangla handwritten compound characters. Azad et al [?] introduce an Autoencoder with a Deep CNN, which they call DConvAENNet for recognizing Bangla Handwritten Character (BHC). A total of 22 experiments were performed on the three-character datasets (BanglaLekha-Isolated, CMATERdb 3.1, Ekush). Their proposed DConvAENNet model achieved 95.21% on BanglaLekha-Isolated for 84 classes, 92.40% on CMATERdb 3.1 for 238 classes, and 95.53% on Ekush for 122 classes. Hossain et al. [?] propose a multi-zoned character segmentation, and a merging method is also proposed, which can produce the handwritten term. Utilizing Convolutional Neural Network (CNN) for preparing 84% precision is accomplished for character level, and 82% precision is achieved in word level.

Although works on multilabel classification in Bangla OCR are yet to be addressed; an instance of a multilabel text classification framework based on a neural network with a simple threshold label predictor has been proposed in [39]. Neural Networks architecture with dropout layer, rectified linear units activation, and AdaGrad optimizer was used in this work with thresh-

olding in multilabel classification to show that the prediction was divided into the true class. In a similar work, Liu et al. [40] presented a CNN classification architecture designed for extreme multilabel recognition of text. The proposed architecture was evaluated on seven different datasets. Analogous to these works depicted here; in this work, a supervised deep convolutional neural network architecture has been developed for Bangla character recognition following fusion multi-single label and multiclass representation of data collected from the several datasets

3 Methodology

Previous works of Bangla character recognition mainly consisted of predicting each of the characters as an individual class focusing on a single label [27, 30, 31, 32, 33, 34, 41]. Some group-based recognition techniques were also attempted by grouping the compound characters based on certain shapes and conditions [35]. In contrast to these existing works, the proposed architecture of multi-labeling of this work is based on the fact that each of the compound characters is a set of multiple consonants where the exact shape of the consonants may not be intact in most of the compound characters. Then those might as well possess the partial shape of the consonants that they are composed of. Thus, if the fragmented and positional shape-based relationships could be established among the consonants of a compound character; then the recognition of that compound character could merely be simplified as consonant character recognition. So consonants and compound characters can be identified using the same set of labeling. And this approach ultimately reduces the number of classes needed to be classified from 210 (171 compounds, 39 consonants) to 39. To incorporate vowels with this we can use them as a separate single-class entity in a single labeling procedure. A simple analogy to this approach is, in the case of multi-class classification, among the predicted output only the maximum value is chosen as the selected one. Which is generally depicted as one hot-coded vector, where every output is labeled zero except only one. If each of the predictions is independent, and the sum of all the predictions are not equal to zero then the following observation can be obtained: If the prediction is chosen to be true or false based on a threshold value. That is, the portion of prediction higher than the threshold value will be chosen as true and if the rest are false, then multiple instances can be predicted by neural networks. Again if only one of the predictions has a higher value than the threshold, the network will act as a single-label multi-class classifier. In this from a classifier only single true

predictions, and multiple true predictions can be obtained. Thus by using a single classifier, consonants, vowels and compound characters can be identified with corresponding single and multi-label classification. The overall workflow is depicted in figure 2.

3.1 Dataset

CMATERdb 3.1.3.2, CMATERdb 3.1.2 (BASIC), Ekush and AI-Bangla, these three different datasets are employed to test and train the model. CMATERdb 3.1.3.2 dataset contains 171 distinct compound characters with 34439 training images and 8520 grey scaled test images. These 171 compound characters represent 171 different classes, where across all classes the train and test image are not equally distributed. This dataset is single labelled. Images in the dataset contain non-centered characters scattered in random positions constituting a difficult problem for pattern recognition. Images of compound characters of this dataset have been used for the purpose of multi-labelling of this work. A total of 368,776 images from the Ekush collection contain 155,570 alphabets, 151,607 compound characters, and 30830 digits. 30769 modifications, as well. Character size affects the image resolution of the Ekush dataset. Most pictures have fewer padding with a white character against a black background. MatrriVasha has put up a dataset that aims to identify Bangla 120 (one hundred twenty) compound characters, which are composed of 2552 (two thousand five hundred fifty-two) isolated handwritten characters authored by different authors and were collected from within Bangladesh. o96

3.2 Consonants and Vowel Labeling

The formation of a uniform labeling or true output for the DL classifier to train and test is the most challenging aspect of the proposed method. Since all of the accessible datasets have a multiclass single-label structure, certain rules that will specify how the true output or label is produced for training and testing datasets must first be defined. The labeling procedure for consonants and vowels is the same as that for single-label processes. A one-hot coded vector format is used for single labeling, where only the output class for each consonant or vowel is set to true (1), while all other classes are set to false (0). As has already been established, a single classifier can do single-label prediction with a threshold value if all but one of the values are below the threshold. The network is thus incentivized to maximize that class and decrease the other classes

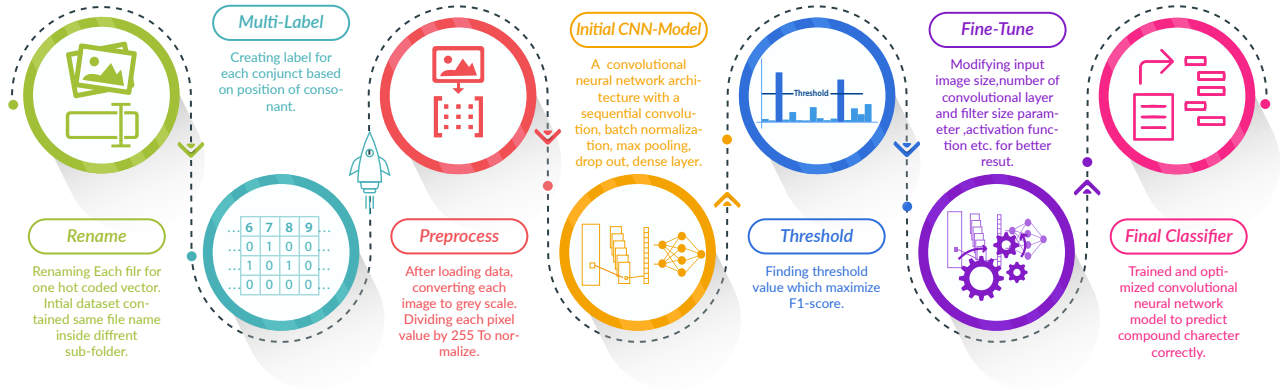


Figure 2: Overview of proposed multi-labelled deep learning based Bangla compound character recognition architecture

by identifying one value as true. The steps for multi-labeling and determining the threshold value are covered in the sections that follow.

Table 1: Standard English equivalent of Bangla compound characters

Bangla character	Corresponding English equivalent
(ক+ষ) = ক্ষ	(k+r)= kr
(ন+ত) = ন্ত	(n+t)=nt
(ত+ন) = ত্ন	(t+n)=tn
(ক+ক) = ক্ক	(k+k)=kk
(ব+ব) = ব্ব	(b+b)=bb
(ঙ+ক) = ক্ঙ	(ng+k)= ngk
(ক+র) = ক্র	(k+r)= kr
(ত+ত+ব) = ত্ত্ব	(t+t+b)=ttb
(দ+ন) = ন্দ	(n+d)=nd
(ব+র) = ব্র	(b+r)=br
(প+র) = প্র	(P+r)=pr
(ক+ত) = ক্ত	(k+t)=kt
(ব+ল) = ব্ল	(b+l)=bl
(স+ক) = স্ক	(s+k)=sk
(স+ত) = স্ত	(s+t)=st
(স+প) = স্প	(s+p)=sp
(ম+প) = ম্প	(m+p)=mp
(স+ত) = স্ত	(t+m)=tm

3.3 Compound Labeling

Most compositions of Bangla compound characters have been incorporated in Bangla from English. Some representations of standard English equivalent of Bangla compound characters shown in table 1. First step of multi-labelling starts with labelling 171 Bangla compound characters. Each of the compound characters is

labelled with those consonants it consists of. For example (ক্ষ) is composed of (ক) and (ষ), then those classes are labelled true for ((ক্ষ)) and other 37 consonants as false. But, there are certain problems with this direct approach of labelling which are as follows:

- There are some compound characters like (ব্ব-ব+ব, ক্ক-ক+ক) consisting of duplicated same consonants.
- Some of the compound characters contain the same set of consonant characters. Owing to the differences in arrangement and position, they represent disparate sets of compound characters. For example (ন্ত) and (ত্ন) both contain same set of consonants ((ন, ত)) but they represent different compound characters.
- Then some of the consonants prevail only in one position in all compound characters. For example (র) appears only in lower portion in all compound characters (প্র, শ্র, ব্র, থ্র), whereas (ঙ) appears only in upper portion (ঙ্গ, ঙ্গ, ক্ঙ).
- Additionally, some of the consonants also embrace fractional facets of the original consonants. For example (ক) appears fractionally in (ক্ষ) whereas intact version of the character is preserved in other compound character like (ক্স).

Now, in order to solve issues stated above, the following multi-labelling approaches have been incorporated in this work:

Each of the compound characters in the dataset is labelled into 78 classes instead of 39, as there are 39 consonants in Bangla. These 78 classes are constituted with the same paired consonants. For example for (ক) instead of (ক) as a single class, there will be (ক1, ক2) two different classes. Thus, now there will be available four labels for any two consonants and four different compound characters of the same set can be la-

belled based on their position, whereas the direct approach would cause labelling of one compound character only. Since, there are at most two compound characters with the same set of consonants available in Bangla compound character dataset, this approach is more feasible than one label per consonant character. For example, if any compound character X consists of two consonants C and D , then there would exist four combination of labelling which is presented as $(C1, D1), (C1, D2), (C2, D1), (C2, D2)$. It has already been stated that there are maximum two compound characters with the same set of consonants in Bangla compound character dataset, so from these label configurations at most two of the labels should be selected. The following figure 3 shows this procedure of pre-calculative labelling.

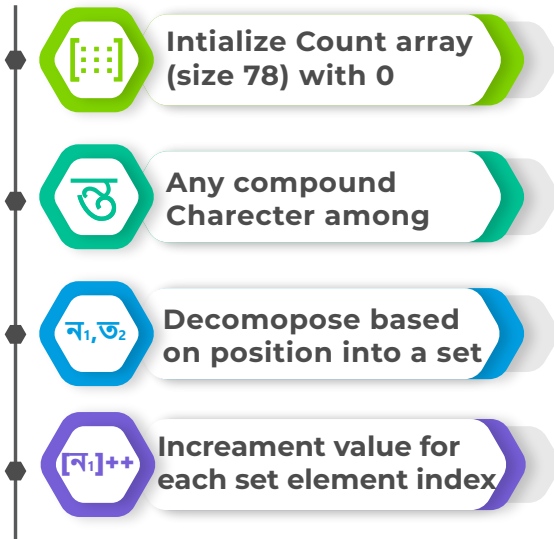


Figure 3: Pre-calculation of multilabelling

Another important issue of multi-labelling is to determine the orders of the labels for each character whether the first or second label is selected. Random permutation of labels for each compound character may lead to erroneous results. One solution could be labelling the consonant characters according to their frequency of appearances in compound characters. The more frequent consonants can be labelled strictly with the first label, then less frequent ones can be labelled based on their position to distinguish different compound characters from the same set of consonants. But this approach should lead to a highly biased dataset. For example for each consonant from (ক) to (ঁ) count of their frequencies is calculated. Of them র appears 30 times in 171 compound character like (ক্র, খ, ঞ, ঞ, চ, ছ, ব, জ, ঞ, ঞ).

For these compound characters র is always labelled as (র1) and other character in those compound characters will be labelled as either character-1 or character-2 like (ক2) in (ক্র) which then shows a contrast to the position based labelling.

Apparently, considering all the issues discussed above; a position based technique based on the placement of the consonants in the compound character has been proposed for this work. This technique is computationally depicted in the algorithm 1. Details of the labelling are as follows:

- So, for those cases where the consonant appears in more than one place in compound character then those consonants should be labelled according to its position. Because of this reason, all consonants in each of the compound characters was pre-calculated based on their position and stored in a vector count. This count vector is the decision making variable for labelling each type of compound character. From a previous example, if any compound character X consists of two equivalent characters C like (বব-ব+ব, ক-ক+ক) then it should be labelled as $(C1, C2)$. Again if two compound characters $X1$ and $X2$ consist of the same set of characters C and D ; if in $X1$, C appears before D [C is high and D is low]; $(C1, D2)$ is chosen as its label and if in $X2$, D appears before C [D is high and C is low]; $(C2, D1)$ is chosen as its label.
- Then for those cases, where the fractional feature of any consonants might appear in one form, it is to be noted that these characters are appearing in a fixed position in those compound characters. For example (ক) appears fractional in (ক্ষ, ক্র), where it can be labelled as (ক2) based on its position of appearance. On the other hand, intact version of those consonants appear in other compound character like (ক) be labelled as (ক1). So, position based labelling also proves to be valid for fractional features of any consonant.
- Now, there are some consonants which appear only in one fixed position should be strictly labelled with the first label.
- As, consonants in compound characters are non-uniformly distributed, some of the consonants never appear in the formation of the compound characters such as (৳, ৳, ৳). Still, labels of these consonants have been incorporated in the list of the labels for generalization and future scalability.

Now, based on the methodology described above for k -th compound character X_k , all labels are stored in one hot coded vector as $0, 0, 0, 1, \dots, 0, 1, 0$. Graphical representation of the proposed multi-labelling approach described above is depicted in figure 4. Then, figure 5

and 6 shows a snapshot of stored hot coded vector of some of compound characters. Due to page restriction, 78 labels have been shown in two separate figures.

Algorithm 1: Proposed Multi label technique

```

Data: Inp, Cmp
Result: Label of all images-L
begin
  initialization;
   $N \leftarrow$  Size of total consonants: 39;
   $L_n \leftarrow$  Number of total labels: 78;
   $L \leftarrow$  one hot coded vector of label, initially all
  zero;
   $C \leftarrow$  Any constant;
   $X \leftarrow$  Any compound character;
   $set \leftarrow$  Set of characters, initially empty;
   $img \leftarrow$  Individual image in dataset;
   $ID \leftarrow$  String contain name of any image;
   $getID(img) \leftarrow$  Returns ID of any image img;
   $high \leftarrow 1$ ;
   $low \leftarrow 0$ ;
   $state(C, X) \leftarrow$  Consonant c position in X high or
  low;
   $st \leftarrow$  bool value high or low , represent consonant
  position;
   $stateSize \leftarrow 2$ ;
   $Inp \leftarrow$  All the image file;
   $Cmp \leftarrow$  list of 171 compound character;
   $Count[N][stateSize] \leftarrow$  Count of the position
  where consonants appear;
   $comp(img) \leftarrow$  returns compound character CRi in
  image img;
   $decompose(X) \leftarrow$  set of consonants in compound
  character X;
  for all compound character  $X_i$  in CR do
     $set \leftarrow decompose(X_i)$ ;
    for all consonants  $C_i$  in set do
       $st \leftarrow state(C_i, X_i)$ ;
       $Count[C_i][st] ++$ ;
    end
  end
  for all the image  $img$  in  $Inp$  do
     $X \leftarrow comp(img)$ ;
     $ID \leftarrow getID(img)$ ;
     $set \leftarrow decompose(X)$ ;
    for all character  $C_i$  in set do
      if  $Count[C_i][1] == 0 // Count[C_i][0] == 0$ 
      then
         $L[ID][c_{i1}] \leftarrow 1$ ;
      end
      else if  $state(C_i, X_i) == high$  then
         $L[ID][c_{i1}] \leftarrow 1$ ;
      else
         $L[ID][c_{i2}] \leftarrow 1$ ;
      end
    end
  end
end

```

3.4 Classification model

A deep Convolutional Neural Network has been utilized for validating the multi-label classification problem. In DCNN, multiple layers are stacked together by using local receptive fields which is based on local connections with sharing weight to ensure better performance and efficiency. The deep architecture implemented for CNN helps to learn diverse and complex features compared to a simple neural network [42]. Thus the primary concept of CNN implies in obtaining local features from input exploiting an image at higher layers subsequently combining them into more compounded features at the lower layers [43]. The particular deep CNN model of this work consists of the following layers:

Convolutional Layer: Convolution layer is the core building block of a Convolutional Neural Network. This layer uses convolution operation to create a feature map for predicting class probabilities of each feature by application of a filter through scanning of the whole image which takes few pixels at a time. The convolution operation is as follows:

$$F(i, j) = (I * K)(i, j) = \sum_m \sum_n I(i + m, j + n)K(m, n) \quad (1)$$

Here I represents the input image, K denotes the 2D filter of size $m \times n$ and F is used to represent the output 2D feature map. This equation shows the convolution of input image I with the filter K (denoted by $I * K$) which subsequently produces the feature map F .

Then, the output of each convolutional layer is fed to an activation function which takes the feature map turned out by the convolutional layer to generate activation map as output. The activation function used in this level is Rectified Linear Unit (RELU) which is based on element wise activation function application mathematically given as the $f(x) = \max(0, x)$ thresholding at zero.

Subsampling (Pooling) Layer: This layer is placed between two convolutional layers which receives several feature maps to subsequently apply the pooling operation to each of them. The pooling operation of this layer reduces the spatial size of the images while reducing the number of parameters in the network; still preserving important characteristics of the images. For this work, max pooling technique has been used which outputs the maximum value in the input region.

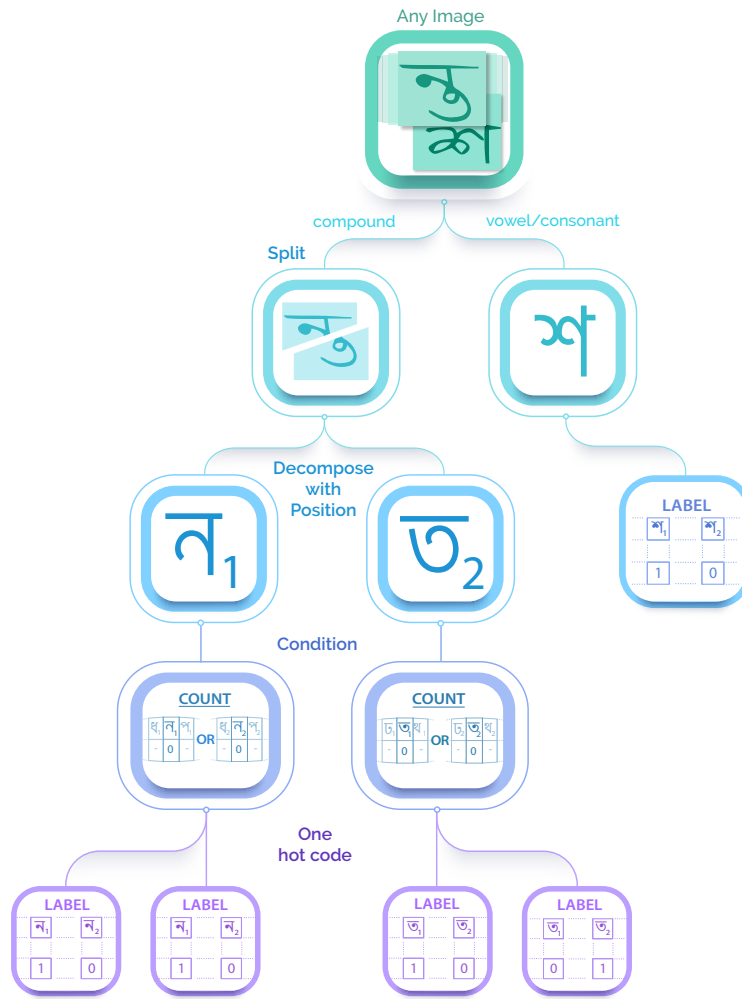


Figure 4: Multi-labelling process of image data based on pre-calculative labelling

Fully Connected Layer: Fully connected input layer contributes to flatten the outputs generated by previous layers by turning them into a single vector. The vector will again be used as an input for the next layer. Then, there comes in the fully connected layer which tends to apply weights over the input generated by the feature analysis for predicting an accurate label. Following that comes in fully connected output layer which generates the final probabilities in order to determine a particular class for each image.

3.4.1 Model Architecture and Development

The deep CNN model of this work is a sequential model with 21 layers consisting of two dimensional convolution, max pooling, dropout and batch normalization. Graphical representation of the classification model is presented in the figure 8.

Image size used from the dataset CMATERdb 3.1.3.2 varies in dimension. For this reason images were resized

at constant dimension for better understanding. Since, high dimensional images require more time to train in the neural network, image size of 94 by 94 pixel excluded of cropping was found optimum for the proposed architecture.

Afterwards in the process of normalizing the images, the images were first converted to a two dimensional vector. Each element of that vector represents pixel value in the image. In this work, both the terms of vector element and pixel value have been used interchangeably. Initially, each of these elements of the vector contains value in range 0 to 255, due to RGB color format. Though convolutional neural networks can operate at any value, it has shown better convergence at 0-1 value. Owing to that reason, RGB value of each image is subsequently divided by 255 to convert those in the range of 0-1 [44].

Now for feature extraction phase; convolution, pooling and batch normalization have been applied subsequently. As, for lower dimensional images, usage of

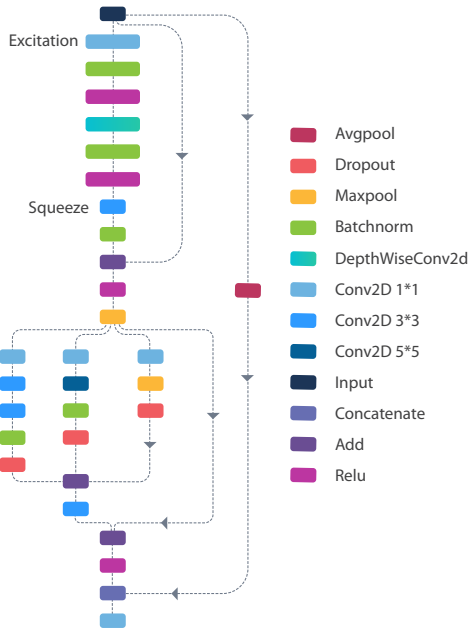


Figure 7: Multilabel Feature Extractor Block (MLFE)

convolutions contains 64,128,128 filters. Batch normalization, max pooling, dropout parameters remain the same. All these layers mentioned above are used for feature extraction. For classification, all the two dimensional values have been flattened and passed into a fully connected dense layer. Two dense layers have been used here with 2048 and 1024 neurons with *ReLU* activation. On the other hand, the output layer contains 78 neurons with *softmax* activation function. Though, in some research it is shown that *sigmoid* activation function works better than *softmax* in multi label classification; in this work, use of *softmax* has provided prominent result [45]. Finally, a dropout value of 0.50 has been used at the end of each dense layer. It is to be noted that, there has been exerted *adam* as optimizer and binary cross entropy as loss function for the proposed architecture. Eventually, total parameters calculated valued to 6,616,590 in total.

The overall classification model formulation procedure is depicted in algorithm 2.

3.5 Thresholding

In single label classification; after the model is trained, the model provides prediction for each label. Labels containing the highest probability tends to be the true class. But, in multilabel classification multiple number of labels can be true and this number can also vary from image to image. For this reason, a threshold value has been used in the proposed classification architecture to

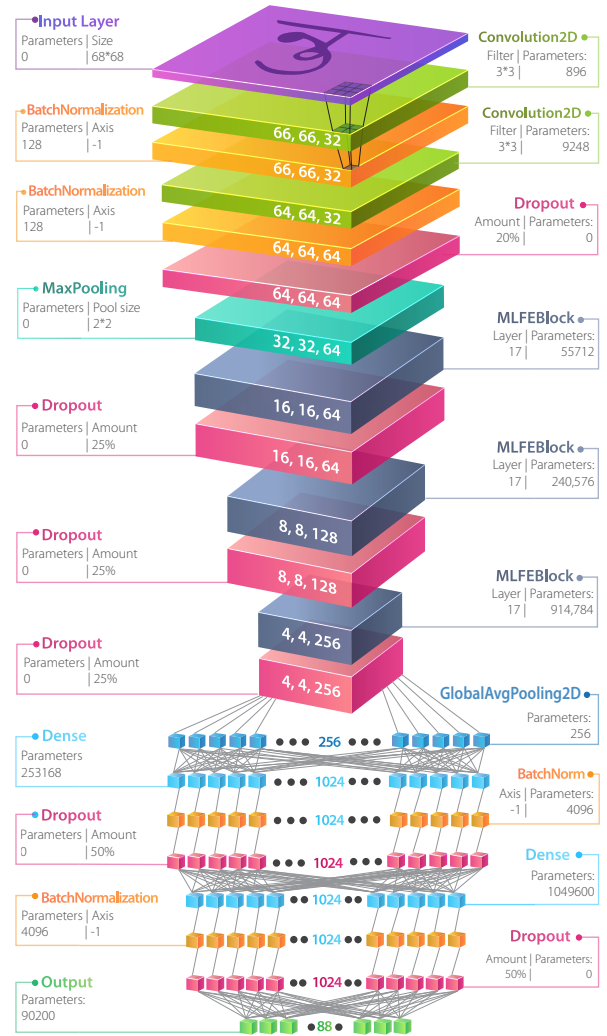


Figure 8: Proposed Deep Convolutional Neural Network model for multilabel classification

determine the correct classes. For each of the labels L_i model produces probability $P(L_i)$. Where,

$$P(L_i) = 1 \quad (2)$$

Thus, for having three true label maximum value of $P(L_i)$ is 0.33 whereas for having two true labels, this value stands at $P(L_i) = 0.5$. A linear search strategy has been exploited in this context in order to find out the threshold value that satisfies the performance requirements for this work. In order to facilitate the threshold finding procedure; evaluation metric F1-scored has been used where for each threshold level starting from from zero; F1-score has been calculated. Every increment of threshold value increases precision and decreases recall. From the calculation procedure from algorithm 3 it has been found out that at threshold level 0.16 F1-score is maximum.

Algorithm 2: Proposed Deep Convolutional Neural Network training model for multi-label classification

```

Data: TrD, TsD
Result: A trained DCNN model
begin
  initialization;
  TrD ← Train Data;
  TsD ← Test Data;
  INP ← 94*94 pixel input image;
  OID ← Output image dimension;
  CVL ← 2D Convolutional Layer;
  BNL ← Batch Normalization Layer;
  MPL ← Max Pooling Layer;
  DOL ← Dropout layer;
  FLT ← Flatten;
  FCL ← Fully Connected Layer;
  OUT ← Output class;
  // Model is initialized as empty sequential model
  for epoch ← 1 to 100 do
    Model.Add_Layer(INP);
    Model.Add_Layer(CVL, ReLU);
    Model.Add_Layer(MPL);
    Model.Add_Layer(DOL);
    while not all layers have been added do
      Model.Add_Layer(CVL, ReLU);
      Model.Add_Layer(BNL);
      Model.Add_Layer(MPL);
      Model.Add_Layer(DOL);
    end
    Model.Add_Layer(FLT);
    Model.Add_Layer(FCL, ReLU);
    Model.Add_Layer(DOL);
    Model.Add_Layer(FCL, ReLU);
    Model.Add_Layer(DOL);
    Model.Add_Layer(OUT, Softmax);
    TrainModel(Model,TrD,TsD);
  end
end

```

Algorithm 3: Threshold value calculation

```

begin
  initialization;
  Min ← 0;
  Max ← 1;
  Thr ← Any threshold value;
  AvgF1(Thr) ← Returns average F1-score of
  dataset for Thr;
  Opthr ← Optimum threshold, initially null;
  MaxF1 ← maximum F1score, initially 0;
  for Thri ← Min to Max do
    if AvgF1(Thri) > MaxF1 then
      MaxF1 = AvgF1(Thri);
      Opthr = Thri;
      Thri = Thri + 0.01;
    end
  end
end

```

4 Experiments and Evaluation

4.1 Experimental Setup

The Experiment is conducted on a machine equipped with intel core i5 processor, 8 GB ram and 2 GB Nvidia 920mx GPU. Keras with tensorflow 2.0 used on the backend with windows operating system. The model is coded with python in the spyder python development environment on anaconda distribution platform. Total 34439 image samples of compound character is used for training, validation is done on 8520 images. For Training and testing, a label container separate “.csv” file is used. To perform computation in a machine with low specification, mini batch with batch size 67 has been used. Following that, two dimensional convolution and subsequent layer described in methodology have been applied accordingly. Fine tuning of various parameters and layers were also conducted to achieve maximum classification performance.

4.2 Evaluation Measures

Performance measures adopted for this work are precision, recall and F1 score [47]. These measures ultimately prove the performance reliability of the proposed multi-labelling technique exploiting the proposed classification architecture. Precision and recall individually can be troublesome to evaluate the model performance owing to the fact that lower threshold value causes higher recall and lower precision and on the other hand, higher threshold value causes lower recall and higher precision. For this reason F1 score has also been used for evaluation to put up a balance between precision and recall. Additionally, micro and macro averaged precision, recall and F1-score have been used as well. Leveraging the fact that micro-averaging emphasizes on common labels so that sparse labels don't impact the the overall model performance being preferred department of multi-label classification problem. At the same time, macro-average method has been used as well to determine the performance of the proposed architecture across over all classes of data [47]. A brief explanation of the evaluation metrics used for this work are as follows:

Precision Precision is that measure system which reveals that how many predictions have been correctly made.

$$Precision = \frac{TP}{TP + FP} \quad (3)$$

TP and FP in equation 3 are True Positive and False Positive respectively.

Recall Measuring the effectiveness of the classifier through positive label retrieval specifies recall.

$$Recall = \frac{TP}{TP + FN} \quad (4)$$

TP and FN in equation 4 are True Positive and False Negative respectively.

F1-score F1-score is the weighted average of Precision and Recall. Overall performance of the proposed architecture can be evaluated better by F1-score as found in the course of study of this work.

$$F1\text{-score} = \frac{2 * (Recall * Precision)}{(Recall + Precision)} \quad (5)$$

Micro-Precision Aggregated contributions of precision of all classes are measured through Micro-precision. In

multi-label classification presence of rare labels must not influence overall precision metric in cases when model performs well on frequent data label. Thus micro-precision has been used as an evaluation metric for this work as it emphasizes on common classes of the dataset.

$$Micro\text{-Precision} = \frac{TP_{sum}}{TP_{sum} + FP_{sum}} \quad (6)$$

Micro-Recall Aggregated contributions of precision of all classes are measured through Micro-precision. In multi-label classification presence of rare labels don't influence overall recall metric in cases when model performs well on frequent data label. Thus micro-recall has been used as an evaluation metric for this work as it emphasizes on common classes of the dataset.

$$Micro\text{-Recall} = \frac{TP_{sum}}{TP_{sum} + FN_{sum}} \quad (7)$$

Micro F1-score Aggregated contribution of all classes by putting up a balance between precision and recall; micro F1-score has been used to measure quality of the proposed multi-label classification problem.

$$Micro\ F1\text{-score} = \frac{2 * (Micro\text{-Recall} * Micro\text{-Precision})}{(Micro\text{-Recall} + Micro\text{-Precision})} \quad (8)$$

Macro-Precision Average precision per class is measured through macro-precision. In multi-label classification overall accuracy of the model is complemented by macro-precision.

$$Macro\text{-Precision} = \frac{1}{N} \sum_{i=0}^N (Precision)_i \quad (9)$$

Here, in equation 9 i is the class index and N the number of classes.

Macro-Recall Average recall per class is measured through macro-precision. In macro averaging, contribution of classes are treated equally regardless of their number presence in the dataset.

$$Macro\text{-Recall} = \frac{1}{N} \sum_{i=0}^N (Recall)_i \quad (10)$$

Here, in equation 10 i is the class index and N the number of classes.

Macro F1-Score In the process of assessing the performance of a model macro F1-score gives equal importance to each label/class.

$$Macro\ F1\text{-score} = \frac{1}{N} \sum_{i=0}^N (F1\text{-Score})_i \quad (11)$$

Here, in equation 11 i is the class index and N the number of classes.

5 Experimental Results and Discussion

Four different models were designed and evaluated in the process of reaching the final model providing best performance. Owing to that, several experiments have been carried out for selecting proper parameters through fine tuning. Different input size, output size, convolution layer, filter size, number of dense layers, dense layer parameters etc have been considered. Each of the models has been trained for 100 epochs. Detail of parameter setting of four different models leading to choose the best model are provided below:

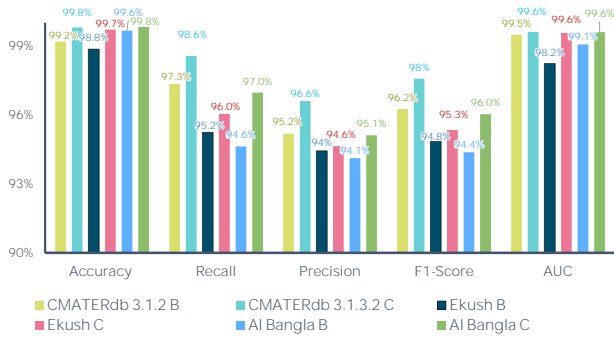


Figure 9: Comparison of different testing performance metrics for 6 datasets

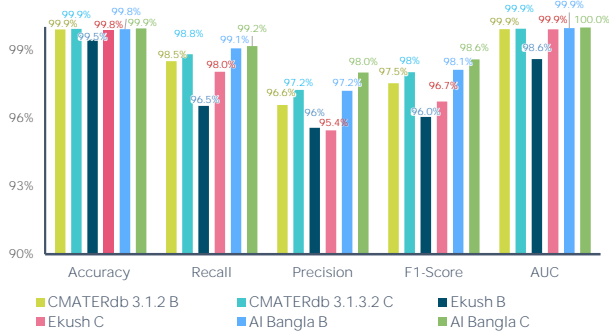


Figure 10: Comparison of different training performance metrics for 6 datasets

5.1 Dataset-Wise

5.2 Hyper Parameter Selection

Model 1 126*126 input image size with five convolution and dropout layer, four max pooling and three batch normalization layer have been incorporated for model 1. Then a total of 32, 64, 128, 128 and 128 number of filters have been used for five convolution layers respectively. Furthermore, max pooling has not been used in the last convolution layer. Image dimension achieved after applying convolution is 4*4 pixels. One hidden dense layer with 2048 neurons has been used before the output layer. Total parameters exploited for this model are 4,745,870. Graphical representation of precision, recall and loss calculation of model 1 are shown in figure 11.

Model 2 94*94 input image size with four convolution and dropout layers, three max pooling and batch normalization layers have been incorporated for model 2. Then a total of 32, 64, 128 and 128 filters have been

used for four convolution layers respectively. Image dimension achieved after applying convolution is 8*8 pixels. Two hidden dense layers with 2048 neurons each have been used before the output layer. Total parameters exploited for this model are 21,326,502. Graphical representation of precision, recall and loss calculation of model 2 are shown in figure 12.

Model 3 94*94 input image size with four convolution, max pooling, dropout layer and three batch normalization layers have been incorporated for model 3. Then a total of 32,64,128,128 filters have been used for four convolution layers respectively. Image dimension achieved after applying convolution is 4*4 pixels. One hidden dense layer with 2048 neuron has been used prior to the output layer. Total parameters exploited for this model are 4,598,286. Graphical representation of precision, recall and loss calculation of model 3 are shown in figure 13.

Model 4 94*94 input image size with four convolution, max pooling, dropout layers and three batch normalization layers have been incorporated for model 4. Then, a total of 32,64,128,128 filters have been used for four convolution layers respectively. Image dimension achieved after applying convolution is 4*4 pixels. Two hidden dense layers with 2048 and 2024 neuron have been used respectively before the output layer. Total parameters exploited for this model are 6,616,590. Graphical representation of precision, recall and loss calculation of model 4 are shown in figure 14.

Table 2 lists accumulated performance measures of all the models designed for this work. Then, following the measurements from the table 2; in the process of choosing the best fitted model, figure 15 shows that the experimented network architecture of model 4 improves the classification performance both in terms of average F1 score with micro and macro averaged F1 score as well. Additionally, figure ?? and ?? shows different F1 measurement of training and validation data for the experimented four models. It is clearly observed from the validation F1 measurement figure that the fluctuation rate of model 4 is visibly less than other model. It represents the fact that model 4 is better trained to provide best classification result. At the same time, model 4 also provides the maximum and steady AUC(micro and macro averaged) 0.97 and 0.96 respectively, found in figure ?? compared to the other AUC(micro and macro averaged) measurements from figure ?? to ??.

Final model selected for this work is thus model 4; detailed structure of which is described in section 3.4.1. Table 3 lists the final optimal hyper-parameter settings corresponding to model 4. This is trained for

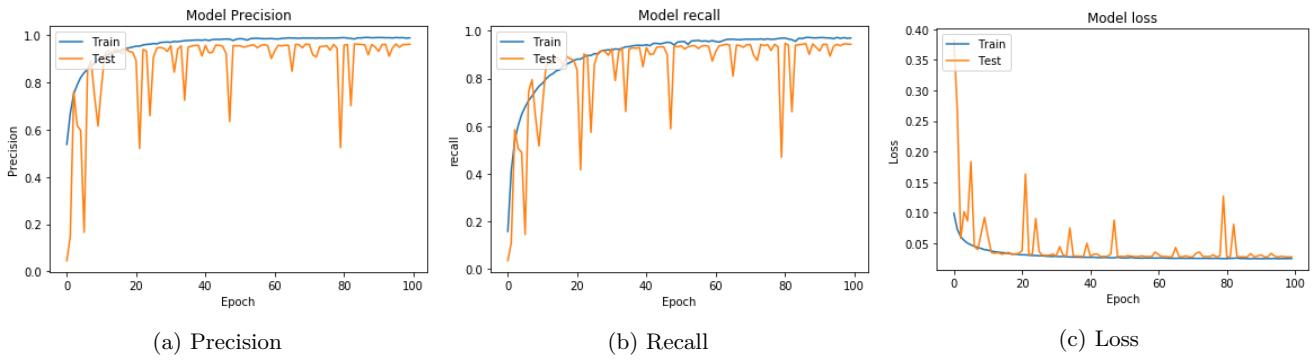


Figure 11: Precision, Recall and Loss measurements of Model 1 of train and test dataset

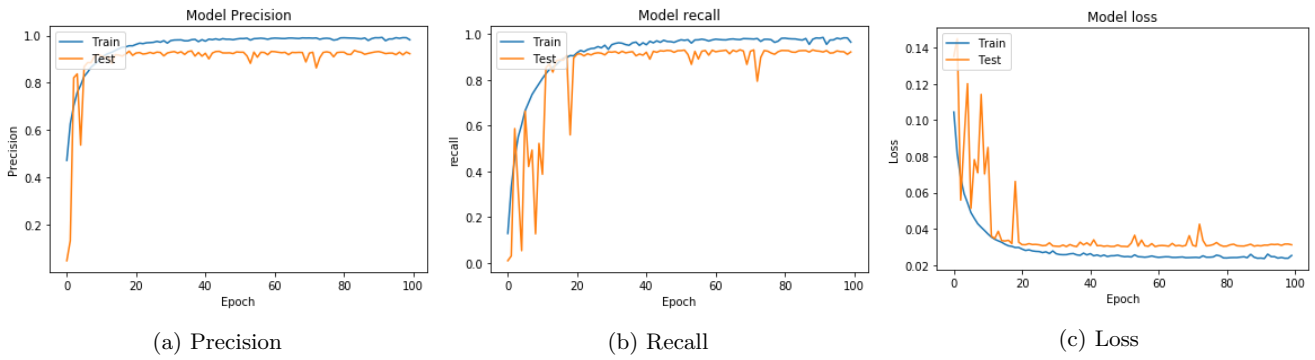


Figure 12: Precision, Recall and Loss measurements of Model 2 of train and test dataset

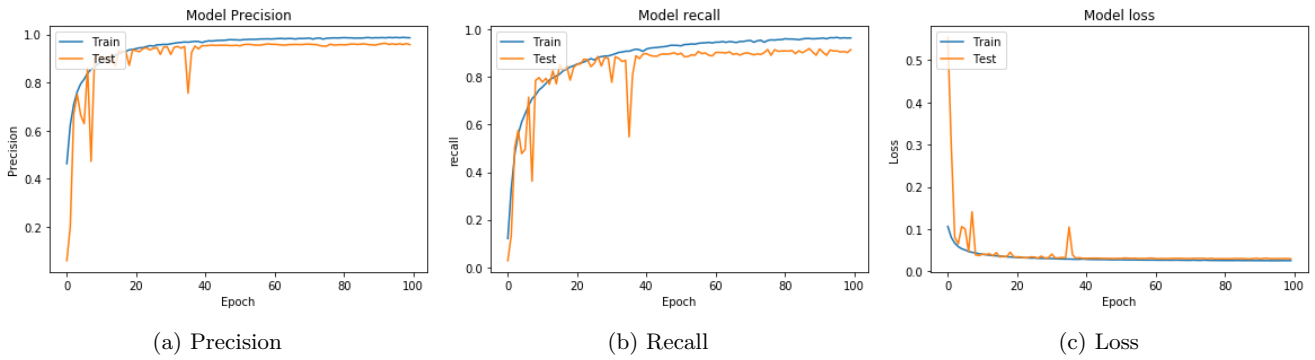


Figure 13: Precision, Recall and Loss measurements of Model 3 of train and test dataset

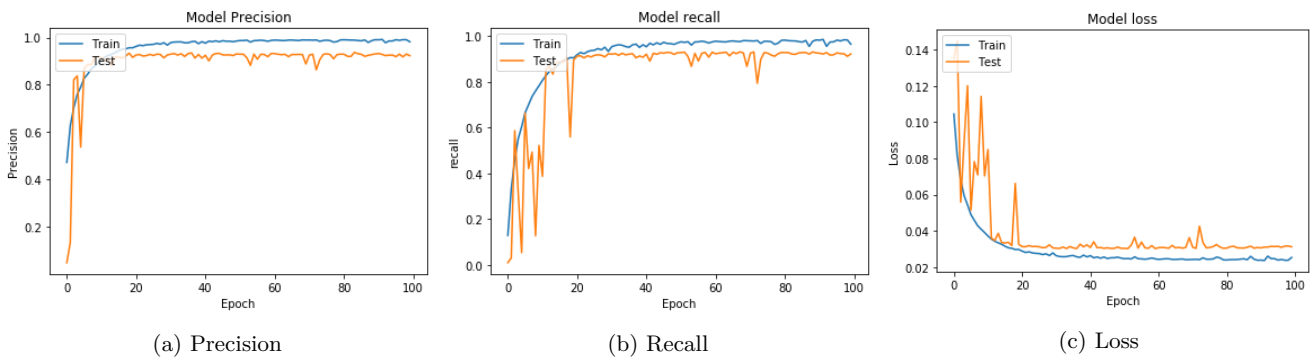


Figure 14: Precision, Recall and Loss measurements of Model 4 of train and test dataset

Table 2: Values of different performance metrics calculated for model 1, 2, 3 and 4

Model	Precision	Recall	F1-score	MacPrec	MacRec	MacF1	MicPrec	MicRec	MicF1
Model 1	≈ 0.90608	≈ 0.89683	≈ 0.90113	≈ 0.90608	≈ 0.89683	≈ 0.90143	≈ 0.93714	≈ 0.92937	≈ 0.93324
Model 2	≈ 0.91156	≈ 0.90792	≈ 0.90860	≈ 0.91156	≈ 0.90792	≈ 0.90973	≈ 0.92304	≈ 0.92219	≈ 0.92262
Model 3	≈ 0.95190	≈ 0.89560	≈ 0.92231	≈ 0.95190	≈ 0.89560	≈ 0.92299	≈ 0.95741	≈ 0.91462	≈ 0.93355
Model 4	≈ 0.91582	≈ 0.9305	≈ 0.92158	≈ 0.91582	≈ 0.9305	≈ 0.9231	≈ 0.93006	≈ 0.93745	≈ 0.93374

Legend: MacPrec–Macro Precision; MacRec–Macro Recall; MacF1–Macro F1; MicPrec–Micro Precision; MicRec–Micro Recall; MicF1–Micro F1

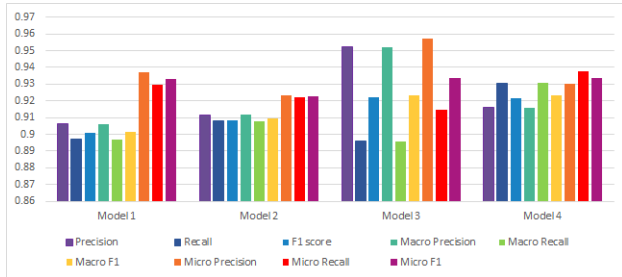


Figure 15: Comparison of different performance metrics for CNN architecture of model 1, model 2, model 3 and model 4

100 epochs. For each epoch training and testing loss, recall, precision, F1 score, macro precision, macro recall, macro F1 score, micro precision, micro recall and micro F1 score have been calculated. For all the epochs, average validation accuracy is 0.979736316, loss is 0.037775678, recall is 0.88655016, precision is 0.894795371, F1 score is 0.88996803. Here accuracy depicts that 97.97% of total validation images is correctly labelled. But the accuracy metric does not give any insight about false classification. For multi-labelling, being able to classify true class is more important. Because of this reason F1 measurement (F1-score) is calculated to describe true class only. Validation curves of model 4 most likely follows training curve for average loss, precision, recall and F1-score as shown in figure 14 and figure ??; which represent the proposed model is neither overfitted nor underfitted. Then, threshold value has also been calculated using the methodology described in section 3.5. The threshold value fixed for the proposed architecture is 0.16, which implies that if probability of any class for any image greater than or equal to 0.16 then this class is meant to be true for that image. Table ?? depicts different recall, precision and F1 measurement score for different threshold value. Figure ?? shows the heatmap corresponding to choose the desired threshold value based on the calculation of maximum F-1 score.

Multi-label classification is the endeavour of selecting multiple category labels from different pre-defined category labels for each class in the dataset. In order to strengthen the claim of better multilabel and

Table 3: Hyper-parameter settings of the proposed deep CNN architecture

Hyper Parameter	Setting
Input Size	94*94
Learning Rate	0.001
Epochs	100
Batch Size	67
Convolution Kernel Size	3*3
Activation Function	ReLU
Batch Normalization Axis	-1
Convolution Dropout Rate	0.25
Maxpool Size	2*2
Number of Hidden Layer	2
Output Activation Function	Softmax
Optimizer	Adam
Loss	Binary Cross Entropy

multiclass performance; classification capability of the proposed architecture has been evaluated using micro and macro averages of precision, recall and F1 score respectively. For all the 100 epochs, average validation micro precision is 0.9300552901, micro recall is 0.9374526566, micro F1 score is 0.9337393226, macro precision is 0.915815347, macro recall is 0.930504688 and macro F1 score is 0.9231015832. A quite standard evaluation score of the micro-averaged F1-score indicates that the proposed classifier performs well on overall data labels and macro-averaged F1-scores shows better performance of the classifier on each individual class.

5.3 Performance comparison of single and multi-label data

To signify the performance of proposed multi-label deep CNN architecture compared to the single label dataset of CMATER db 3.1.3.2, a single label model is also designed. The only difference of the single label deep CNN model to that of proposed multi-label deep CNN model is that sparse categorical cross entropy has been used as loss function. Other than that, input image dimension, number of convolution layer, dense layer and convolution filter size are all identical between multi-label and single label data based architecture. In order to evaluate the performance of the multi-label deep CNN architecture; accuracy of the classifier has been evaluated

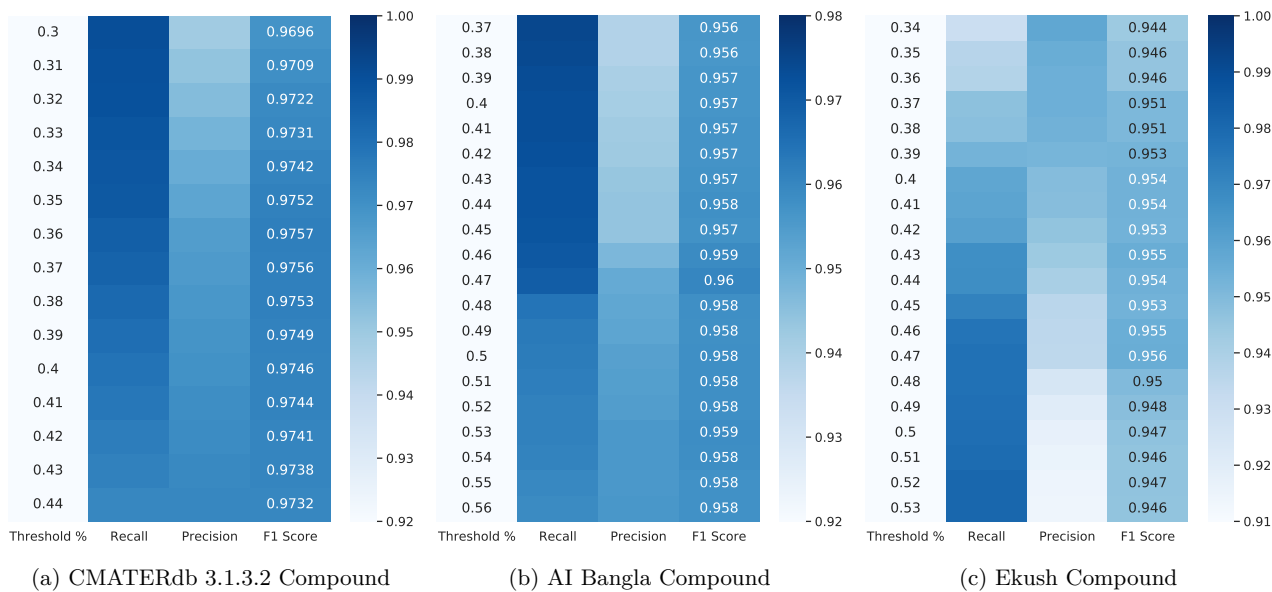


Figure 16: Heatmap shows that maximum performance metric in terms of F1-score has been achieved with each threshold value

with that of the single label deep CNN model. From there, training and validation accuracy of the proposed multi-label classifier stands almost 98% and 97%; on the contrary training and validation accuracy of the single label classifier stands almost 82% and 78.5%. Figure 17 shows the training set and validation set of accuracy(recognition rate) between proposed multi-label model and single label classification model. This accuracy depiction clearly shows the superiority of multi-label depiction of dataset over single label dataset exploiting identical classifier.

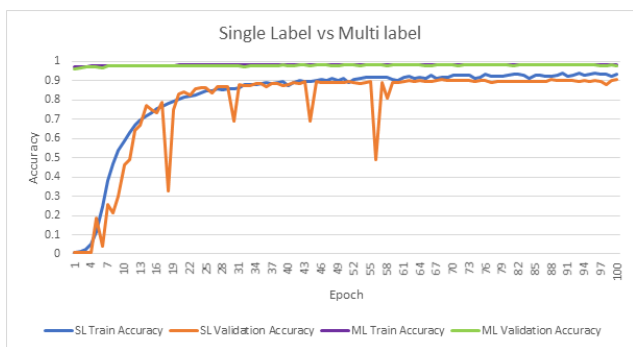


Figure 17: Accuracy comparison of train and test dataset of proposed dataset architecture to single label dataset architecture

5.4 Test with real data excluded of dataset

In order to ensure the efficacy of the proposed multi-label deep CNN architecture, images of some randomly collected compound characters have been separately investigated as well. Then consecutive prediction has been calculated based on the proposed trained model. Images of written implication of Bangla compound characters and excluded from dataset have been used for this work. This test has been conducted to ensure the ramifications of threshold on the proposed classification architecture. Figure 18 shows that the classifier predicts those compound characters which are not from the dataset. Probability of each class is represented as bar diagram. True class stated here represents correct label for each image in terms of prediction bar diagram. Probability value greater than or equal threshold value is ought to be true class.

In view of lots of variation in handwritten script, some of the images were misclassified as well. One of the common reasons for misclassification is that, true class prediction is little less than threshold value though in sorted descending order those class also account for greater performance measure, such as (অ, ভূ, ক). Again, there are also found some compound characters which predict more classes than those which it is composed of which meant to be true classes as well owing to prediction value greater than threshold value. Following that, this scenario of (স, ব, জ, ক, গ) predicted classes contain true classes with some other false classes. Some



Figure 18: Compound character prediction with true class depiction

of the misclassified compound characters are shown in figure 19.

6 Conclusion

In this research work, a multi-labelling technique for Bangla characters has been proposed with a DCNN classifier for corresponding character recognition. Reduction of classes has been achieved through multi-labelling ultimately commuting the complexity of recog-

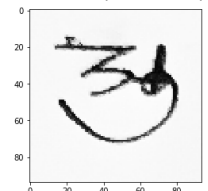
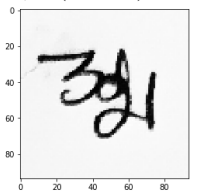
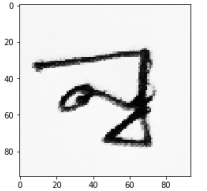
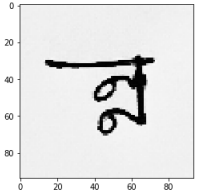
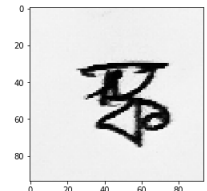
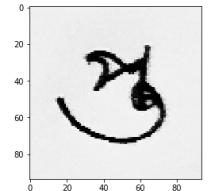
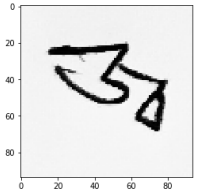
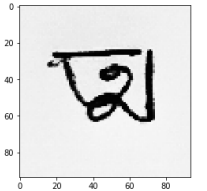
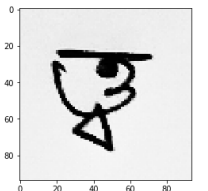
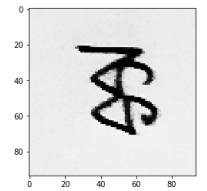
<p>True class : 30,54 (স1,ত1) Predicted as: 15,30,54 (ত1,স1,ত2)</p> 	<p>True class : 30,55 (স1 থ2) Predicted as: 30,63 (স1, ম2)</p> 	<p>True class : 19,61 (ন1,ব2) Predicted as: 16,24,61 (থ1,ম1,ব2)</p> 	<p>True class : 19,58 (ন1, ন2) Predicted as: 19,63,58 (ন1,ন2,ন2)</p> 	<p>True class : 4,39 (ঙ1, ক2) Predicted as: 39 (ক2)</p> 
<p>True class : 20,54 (প1,ত2) Predicted as: 15,20,30,54(ত1,প1,স1,ত2)</p> 	<p>True class : 7,61 (জ1,ব2) Predicted as: 7,46,61 (জ1,জ2,ব2)</p> 	<p>True class : 15,63 (ত1,ম2) Predicted as: 15 (ত1)</p> 	<p>True class : 15,54,61 (ত1,ত2,ব2) Predicted as: 15,61 (ত1,ব2)</p> 	<p>True class : 0,39 (ক1,ক2) Predicted as: 0,10,39 (ক1,ট1,ক2)</p> 

Figure 19: Misclassification occurrences

dition. The process of compound character recognition has been transformed into recognition of those consonants which constitute a particular compound character based on their adjoining relationship. Consequently this aspect implies that separate representation of classes for compound character recognition is no longer needed. Several performance metrics have been evaluated for this work. The experimental result also indicates that DCNN architecture provides satisfactory results for the proposed multi-labelled representation of data. Most importantly, this work can be utilized as a basis for future apprehension of multi-labelling for compound characters in Bangla OCR. Finally, application of more advanced neural network model for depicting the proposed multi-labelling technique would constitute our future perspective by achieving 100% recognition rate.

Declarations

Funding: N/A

Conflict of Interest Statement: The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

Availability of data and material: The dataset analysed during the current study are available in the CMA-TERdb 3.1.3.2 repository, available at: <https://code.google.com/archive/p/cmaterdb/downloads#makechanges>

Code availability: Code of the proposed classifier will be made available once this work gets accepted.

Authors and Contributors: This work has been carried out in close collaboration between all co-authors. All authors have contributed to, seen and approved the final manuscript.

References

1. Singh A, Bacchuwar K, Bhasin A. A Survey of OCR Applications. *International Journal of Machine Learning and Computing (IJMLC)*. 2012 01.
2. Nagy G. At the frontiers of OCR. *Proceedings of the IEEE*. 1992;80(7):1093-100.
3. Srihari SN, Shekhawat A, Lam SW. In: *Optical Character Recognition (OCR)*. GBR: John Wiley and Sons Ltd.; 2003. p. 1326–1333.
4. Heutte L, Paquet T, Moreau JV, Lecourtier Y, Olivier C. A Structural/Statistical Feature Based Vector for Handwritten Character Recognition. *Pattern Recogn Lett*. 1998 May;19(7):629–641. Available from: [https://doi.org/10.1016/S0167-8655\(98\)00039-7](https://doi.org/10.1016/S0167-8655(98)00039-7).
5. Ganis MD, Wilson CL, Blue JL. Neural network-based systems for handprint OCR applications. *IEEE Transactions on Image Processing*. 1998;7(8):1097-112.
6. Singh A, Bacchuwar K, Choubey A, Kumar D, Karanam S. An OMR based automatic music player. *ICCRD2011 - 2011 3rd International Conference on Computer Research and Development*. 2011 03;1.
7. Gossweiler R, Kamvar M, Baluja S. What's up CAPTCHA?: a CAPTCHA based on image orientation. In: *WWW 2009*; 2009. p. 841-50.
8. Barwick J. Building an institutional repository at Loughborough University: Some experiences. *Program: electronic library and information systems*. 2007 05;41.

9. Indira K, Mohan K, Nikhilashwary T. 8. In: Automatic License Plate Recognition. Springer Singapore; 2019. p. 67-77.
10. Shyang-Lih Chang, Li-Shien Chen, Yun-Chung Chung, Sei-Wan Chen. Automatic license plate recognition. *IEEE Transactions on Intelligent Transportation Systems*. 2004;5(1):42-53.
11. Yuan A, Bai G, Jiao L, Liu Y. Offline Handwritten English Character Recognition Based on Convolutional Neural Network. *Proceedings - 10th IAPR International Workshop on Document Analysis Systems, DAS 2012*. 2012 03.
12. Das RL, Prasad BK, Sanyal G. HMM based Offline Handwritten Writer Independent English Character Recognition using Global and Local Feature Extraction. *International Journal of Computer Applications*. 2012;46:45-50.
13. Taft M, Zhu X, Peng D. Positional Specificity of Radicals in Chinese Character Recognition. *Journal of Memory and Language - J MEM LANG*. 1999 05;40:498-519.
14. Liu C, Yin F, Wang D, Wang Q. Chinese Handwriting Recognition Contest 2010. In: 2010 Chinese Conference on Pattern Recognition (CCPR); 2010. p. 1-5.
15. Zhou X, Yu J, Liu C, Nagasaki T, Marukawa K. Online Handwritten Japanese Character String Recognition Incorporating Geometric Context. In: Ninth International Conference on Document Analysis and Recognition (ICDAR 2007). vol. 1; 2007. p. 48-52.
16. Amin A. Off-line Arabic character recognition: the state of the art. *Pattern Recognition*. 1998;31(5):517-530. Available from: <http://www.sciencedirect.com/science/article/pii/S0031320397000848>.
17. Amin A, Al-Sadoun H, Fischer S. Hand-printed arabic character recognition system using an artificial network. *Pattern Recognition*. 1996;29(4):663-675. Available from: <http://www.sciencedirect.com/science/article/pii/0031320395001107>.
18. Gunawan D, Arisandi D, Ginting FM, Rahmat RF, Amalia A. Russian Character Recognition using Self-Organizing Map. *Journal of Physics: Conference Series*. 2017 jan;801:012040. Available from: <https://doi.org/10.1088%2F1742-6596%2F801%2F1%2F012040>.
19. Das N, Acharya K, Sarkar R, Basu S, Kundu M, Nasipuri M. A benchmark image database of isolated Bangla handwritten compound characters. *International Journal on Document Analysis and Recognition (IJ DAR)*. 2014 Dec;17(4):413-31. Available from: <https://doi.org/10.1007/s10032-014-0222-y>.
20. Pal U, Wakabayashi T, Kimura F. Handwritten Bangla Compound Character Recognition Using Gradient Feature. In: 10th International Conference on Information Technology (ICIT 2007); 2007. p. 208-13.
21. Das N, Acharya K, Sarkar R, Basu S, Kundu M, Nasipuri M. A Novel GA-SVM Based Multistage Approach for Recognition of Handwritten Bangla Compound Characters. In: Satapathy SC, Avadhani PS, Abraham A, editors. *Proceedings of the International Conference on Information Systems Design and Intelligent Applications 2012 (INDIA 2012) held in Visakhapatnam, India, January 2012*. Berlin, Heidelberg: Springer Berlin Heidelberg; 2012. p. 145-52.
22. Krizhevsky A, Sutskever I, Hinton GE. ImageNet classification with deep convolutional neural networks. *Communications of the ACM*. 2017 May;60(6):84-90. Available from: <https://doi.org/10.1145/3065386>.
23. Garain U, Chaudhuri BB. Segmentation of touching characters in printed Devnagari and Bangla scripts using fuzzy multifactorial analysis. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*. 2002;32(4):449-59.
24. Chaudhuri BB, Pal U. A complete printed Bangla OCR system. *Pattern Recognit*. 1998.
25. Majumdar A. Bangla Basic Character Recognition Using Digital Curvelet Transform. *Journal of Pattern Recognition Research*. 2007 01;1:17-26.
26. Bhowmik T, Ghanty P, Roy A, Parui S. SVM-based hierarchical architectures for handwritten Bangla character recognition. *Document Analysis and Recognition*. 2009 06;12:97-108.
27. Das N, Sarkar R, Basu S, Saha P, Kundu M, Nasipuri M. Handwritten Bangla character recognition using a soft computing paradigm embedded in two pass approach. *Pattern Recognition*. 2015 06;48.
28. Minaee S, Boykov YY, Porikli F, Plaza AJ, Kehtarnavaz N, Terzopoulos D. Image segmentation using deep learning: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*. 2021.
29. Wei Y, Xia W, Huang J, Ni B, Dong J, Zhao Y, et al. CNN: Single-label to multi-label. *arXiv preprint arXiv:14065726*. 2014.
30. Das N, Basu S, Sarkar R, Kundu M, Nasipuri M. Handwritten Bangla Compound character recognition: Potential challenges and probable solution.; 2009. p. 1901-13.
31. Alom MZ, Sidike P, Hasan M, Taha TM, Asari VK. Handwritten Bangla Character Recognition Using The State-of-Art Deep Convolutional Neural Networks. *CoRR*. 2017;abs/1712.09872. Available from: <http://arxiv.org/abs/1712.09872>.
32. Rahman M, Akhand MAH, Islam S, Shill P, Rahman MM. Bangla Handwritten Character Recognition using Convolutional Neural Network. *International Journal of Image, Graphics and Signal Processing (IJIGSP)*. 2015 07;7:42-9.
33. Das N, Das B, Sarkar R, Basu S, Kundu M, Nasipuri M. Handwritten Bangla Basic and Compound character recognition using MLP and SVM classifier. *Journal of Computing*. 2010 02;2.
34. Roy S, Das N, Kundu M, Nasipuri M. Handwritten isolated Bangla compound character recognition: A new benchmark using a novel deep learning approach. *Pattern Recognition Letters*. 2017;90:15-21. Available from: <http://www.sciencedirect.com/science/article/pii/S0167865517300703>.
35. Pramanik R, Bag S. Shape Decomposition-based Handwritten Compound Character Recognition for Bangla OCR. *Journal of Visual Communication and Image Representation*. 2017 11;50:123-34.
36. Ferdous J, Karmaker S, Rabby ASA, Hossain SA. MatriVasha: a multipurpose comprehensive database for Bangla handwritten compound characters. In: *Emerging Technologies in Data Mining and Information Security: Proceedings of IEMIS 2020, Volume 3*. Springer; 2021. p. 813-21.
37. Hasan MM, Abir MM, Ibrahim M, Sayem M, Abdullah S. Aibangla: A benchmark dataset for isolated bangla handwritten basic and compound character recognition. In: 2019 International Conference on Bangla Speech and Language Processing (ICBSLP). IEEE; 2019. p. 1-6.
38. Rabby ASA, Haque S, Islam MS, Abujar S, Hossain SA. Ekush: A multipurpose and multitype comprehensive database for online off-line bangla handwritten characters. In: *International Conference on Recent Trends in Image Processing and Pattern Recognition*. Springer; 2018. p. 149-58.
39. Nam J, Kim J, Gurevych I, Fürnkranz J. Large-scale Multi-label Text Classification - Revisiting Neural Networks. *CoRR*. 2013;abs/1312.5419. Available from: <http://arxiv.org/abs/1312.5419>.

40. Liu J, Chang WC, Wu Y, Yang Y. Deep learning for extreme multi-label text classification. In: Proceedings of the 40th international ACM SIGIR conference on research and development in information retrieval; 2017. p. 115-24.
41. Mashrukh Zayed M, Neyamul Kabir Utsha SM, Waheed S. Handwritten Bangla Character Recognition Using Deep Convolutional Neural Network: Comprehensive Analysis on Three Complete Datasets. In: Kaiser MS, Bandyopadhyay A, Mahmud M, Ray K, editors. Proceedings of International Conference on Trends in Computational and Cognitive Engineering. Singapore: Springer Singapore; 2021. p. 77-87.
42. Albawi S, Mohammed TA, Al-Zawi S. Understanding of a convolutional neural network. In: 2017 International Conference on Engineering and Technology (ICET); 2017. p. 1-6.
43. Wan J, Wang D, Hoi SCH, Wu P, Zhu J, Zhang Y, et al. Deep Learning for Content-Based Image Retrieval: A Comprehensive Study. In: Proceedings of the 22nd ACM International Conference on Multimedia. MM '14. New York, NY, USA: Association for Computing Machinery; 2014. p. 157-166. Available from: <https://doi.org/10.1145/2647868.2654948>.
44. Finlayson GD, Schiele B, Crowley JL. Comprehensive colour image normalization. In: Burkhardt H, Neumann B, editors. Computer Vision — ECCV'98. Berlin, Heidelberg: Springer Berlin Heidelberg; 1998. p. 475-90.
45. Liu J, Chang WC, Wu Y, Yang Y. Deep Learning for Extreme Multi-Label Text Classification. In: Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval. SIGIR '17. New York, NY, USA: Association for Computing Machinery; 2017. p. 115-124. Available from: <https://doi.org/10.1145/3077136.3080834>.
46. Ioffe S, Szegedy C. Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. CoRR. 2015;abs/1502.03167. Available from: <http://arxiv.org/abs/1502.03167>.
47. Herrera F, Charte F, Rivera Rivas A, Del Jesus MJ. Multilabel Classification. Springer International Publishing; 2016.