Noname manuscript No.
(will be inserted by the editor)

Optical character recognition using deep learning techniques for printed and handwritten documents.

Sanika Bagwe · Vruddhi Shah · Jugal Chauhan · Purvi Harniya · Amanshu Tiwari · Vartika Gupta · Durva Raikar · Vrushabh Gada · Urvi Bheda · Vishant Mehta · Mahesh Warang · Ninad Mehendale

Received: date / Accepted: date

Abstract Despite decades of research, developing optical character recognition (OCR) systems with capabilities comparable to that of a human remains an open challenge. A large scale of documents in the form of the image is needed to be entered into computer databases which takes a lot of memory as compared to editable text and there can be errors while interpretation of data from an image. This project aims to use OCR to convert handwritten or printed documents into editable text. Documents are scanned to image format as an input to a doc_class_net which is a full size image classifier that classifies the input image into four different classes viz. printed, semi-printed, handwritten discrete, and handwritten cursive. The OCR model predicts and then decodes the text in the image and gives the output as an editable text. We have applied OCR to printed text images using the Pytesseract. For handwritten text images, the text is predicted using a self-developed convolutional recurrent neural network (CRNN) named CL-9 (7 CNN layers and 2 LSTM layers). The accuracy of the doc_class_net classifier and line_class_net classifier(line wise classifier) was 88.03 % and 82.1 % respectively. The overall accuracy for printed, handwritten discrete and handwritten cursive obtained is 94.79 %, 75.2 %, and 65.7 % respectively. OCR has real-time applications in various fields like medical prescriptions, smart libraries, and tax returns. Using this method books, magazines, and any other form of documents can be digitized and made accessible very efficiently.

Keywords OCR for cursive handwriting · Document type classifier · Line OCR

1 Introduction

In a society that is now digitally enhanced, we depend on computers to process huge amounts of data. Various economic and business requirements demand a fast inputting of huge volumes of data into the computers. This cannot be achieved by manually typing the data and entering it into the computers as it is very time-consuming. Hence mechanizing the manual process plays an important role. Many kinds of research came in the character recognition area where optical character recognition (OCR) has made a mark. Optical character recognition is known as the process of reading the text from the documents, both the printed text and handwritten text and converting the text into a form that the computers can operate on.

Optical character recognition is the translation of handwritten, typewritten, or printed paper into machine editable text by using any scanning device or software. It is a field of research in pattern recognition, machine vision, and artificial intelligence. And each year, this technology helps us free large amounts of physical storage space once given over to file cabinets and boxes of paper documents.

In our manuscript, we have proposed a model that uses Pytesseract and CRNN to convert scanned images of input documents into machine editable text. Neural networks learn and remember what they have learned, enabling them to predict classes or values for new datasets, but what makes CRNN different is that unlike normal neural networks, CRNNs rely on the information from previous output to predict for the up-

* Corresponding author
N. Mehendale
B-412, K. J. Somaiya College of Engineering, Mumbai, India
Tel.: +91-9820805405
E-mail: ninad@somaiya.edu

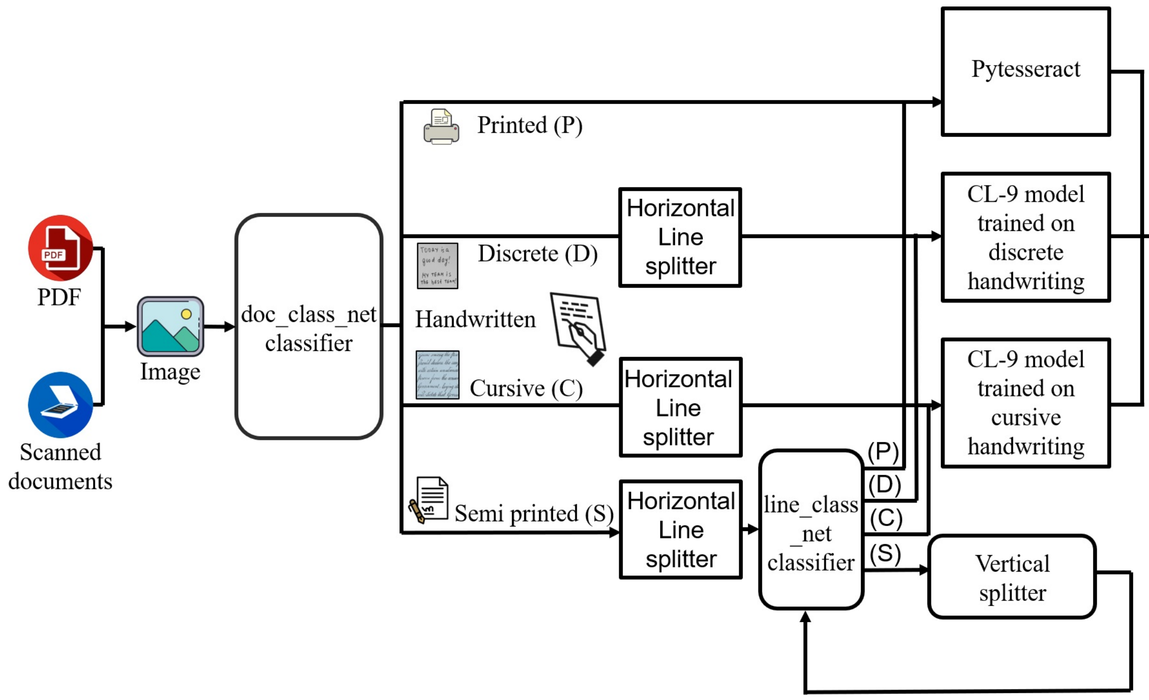


Fig. 1 Proposed system flow diagram. The raw image being fed to the doc_class_net classifier which classifies the input into four classes namely printed, handwritten discrete, handwritten cursive, and semi printed. If the image is classified as printed it is passed to Pytesseract. Images classified as handwritten cursive or handwritten discrete are passed to horizontal line splitter. The line-wise images are then passed to the CL-9 model trained to predict cursive and discrete text. If an image is classified as semi printed, it is passed to a horizontal line splitter and then the line-wise images are passed to the line_class_net classifier. The line_class_net classifier further segregates the line image into printed, semi-printed, handwritten discrete and handwritten cursive. If the output is printed, handwritten cursive, or handwritten discrete, we send the image to the respective CL-9 model and if the output is semi printed, it is passed through a vertical splitter. The split images are fed again to the line_class_net classifier and the process is repeated until there's no semi printed image.

coming input. Firstly, the input documents are converted into an image format, which is then classified into printed, handwritten cursive, handwritten discrete and semi-printed by an input classifier. Text in the image is predicted using corresponding models. The predicted text is therefore available as machine editable text which can be retrieved easily whenever necessary.

2 Literature review

Multiple research works have been carried out in the field of OCR. Handwritten characters are much more difficult to recognize than the printed characters due to the difference in writing styles for different people. The OCR method has been used in converting printed text into editable text. OCR is a very useful and popular method in various applications. Accuracy of OCR can be dependent on text preprocessing and segmentation algorithms [1]. A classifier for predicting the character accuracy was achieved by Blando *et al.* [2] using the OCR system on a given page. This classifier was based on measuring the amount of white speckle, the number of character fragments, and overall size information on the page. Results of processing 639 pages show a recognition rate of approximately 85 %. A novel method for training RNNs to label unsegmented sequences directly, thereby solving the problems of pre-segmented training data, and post-processing to transform their outputs into label sequence was done by Graves *et al.* [3]. An experiment on the TIMIT speech corpus demonstrated its advantages over both a baseline HMM and a hybrid HMM-RNN. Deep learning techniques such as convolutional neural networks can show better accuracy than conventional techniques. You *et al.* [4] proposed the concept of detection of figure-panel labels in medical journal articles using Markov Random Field. The input data was preprocessed and then using OCR, the result was classified as a figure-panel label or a noise with post-processing. Mariappan *et al.* [5] designed a method to acquire the multiple medical instruments' LCD reading using the concept of the virtual medical instrument based on LabVIEW and its optical character recognition (OCR) module. The data obtained was then transmitted to OTOROB's remote computer over the internet. The OCR system was trained continuously until it was able to recognize characters consistently. Kumar *et*

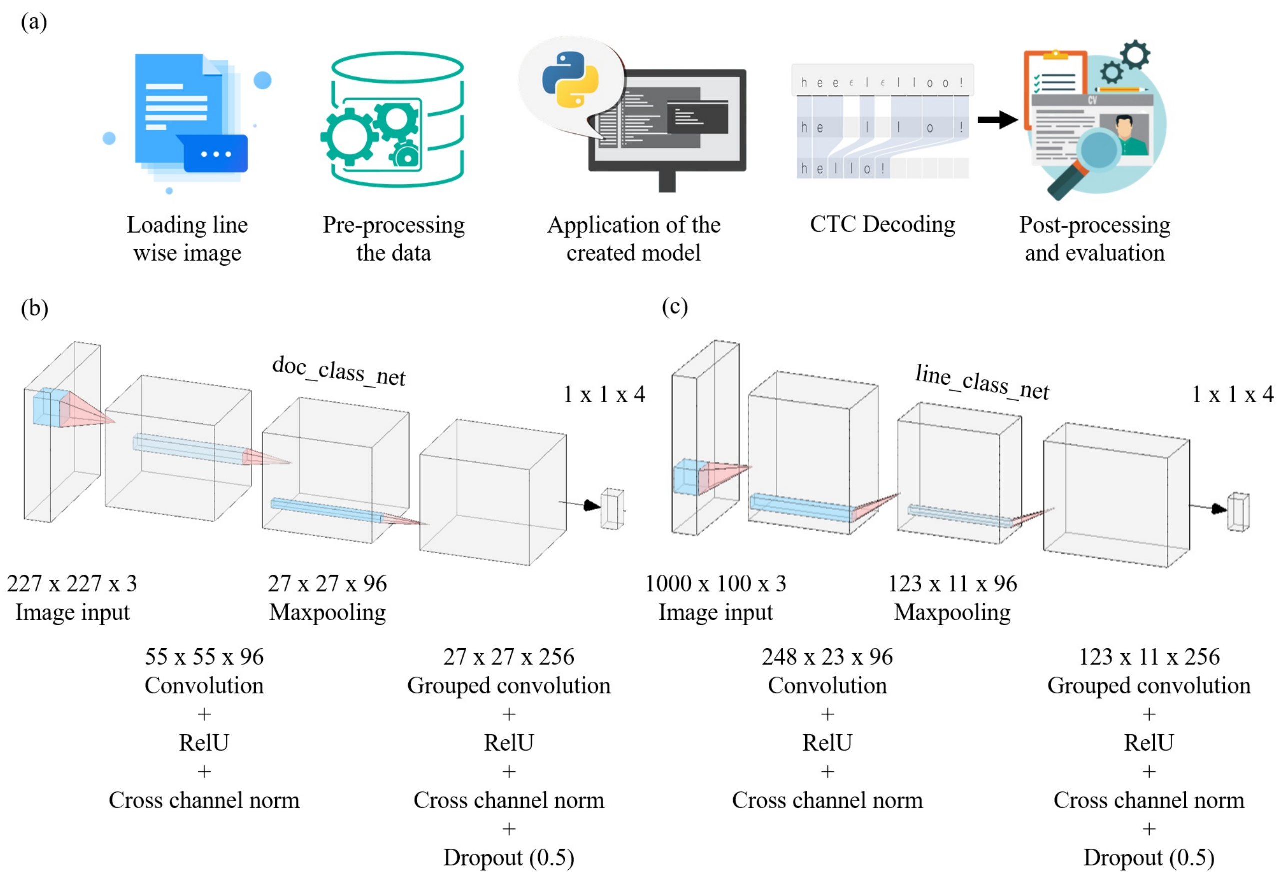
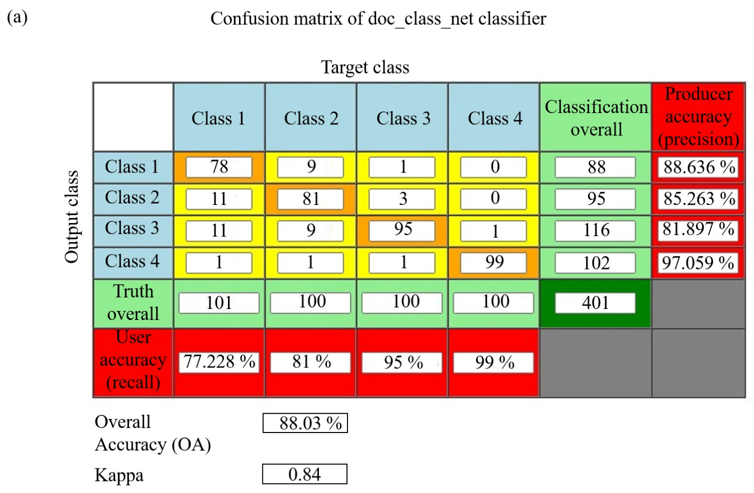

(a)
Loading line wise image
Pre-processing the data
Application of the created model
h e e ϵ l ϵ l l o o !
h e l l o !
h e l l o !
CTC Decoding
CV
Post-processing and evaluation
(b)
doc_class_net
1 x 1 x 4
227 x 227 x 3 Image input
27 x 27 x 96 Maxpooling
55 x 55 x 96 Convolution + RelU + Cross channel norm
27 x 27 x 256 Grouped convolution + RelU + Cross channel norm + Dropout (0.5)
(c)
line_class_net
1 x 1 x 4
1000 x 100 x 3 Image input
123 x 11 x 96 Maxpooling
248 x 23 x 96 Convolution + RelU + Cross channel norm
123 x 11 x 256 Grouped convolution + RelU + Cross channel norm + Dropout (0.5)

Fig. 2 (a) The line-wise image is preprocessed first. The trained CL-9 model predicts the text which is decoded using a CTC beam search decoder. In the post-processing step autocorrect is applied and the evaluation is done. (b) The architecture of the doc_class_net. It consist of (227 x 227 x 3) input image passed to (55 x 55 x 96) convolution + RelU + Cross channel norm layer which is further connected to a (27 x 27 x 96) Maxpooling layer followed by a (27 x 27 x 256) Grouped convolution + RelU + cross channel norm + Dropout (0.5) layer and finally (1x1x4) layer at the output. (c) The architecture of the line_class_net. It consists of a (1000 x 1000 x 27) input image passed to a(248 x 23 x 96) Convolution+RelU+ Cross channel norm layer followed by a (123 x 11 x 96) Maxpooling layer followed by a (123 x 11 x 256) Grouped convolution + RelU + Cross channel norm + Dropout(0.5) layer and finally (1 x 1 x 4) layer at the output.

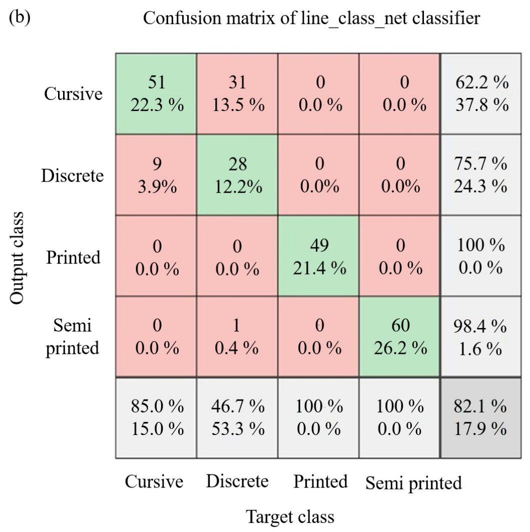
al. [6] used the techniques of OCR for offline handwriting recognition of the doctors and were able to detect and analyze the handwriting words. They tried to get some pattern in the doctor's handwriting which was used to guess the most probable prescribed medicine. A weakly-supervised framework for action labeling in the video was designed by Huang *et al.* [7]. They introduced the extended connectionist temporal classification (ECTC) framework to efficiently evaluate all possible alignments via dynamic programming and explicitly enforce their consistency with frame-to-frame visual similarities. This protected the model from distractions of visually inconsistent or degenerated alignments without the need for temporal supervision. Two mainstream methods adopted for text recognition, tweaking the tesseract pipeline for improving the existing results, and using a single shot multibox detector for segmenting the text regions and training it on the synthetically generated annotated data was done by Mithal *et al.* [8]. Weng *et al.* [9] used a machine learning-based natural language processing approach, where the concept of natural language processing along with content derived meta-data was used in developing the machine learning pipeline and to develop medical subdomain classifiers based on the content of the note. Usually, the OCR methods are designed for single-scale characters, which tend to have poor performance in complex scenarios. Scheidl *et al.* [10] proposed a word beam search decoding technique. It constrains words to those contained in a dictionary that allows arbitrary non-word character strings between words and has a better running time than token passing. Based on Convolutional Recurrent Neural Network (CRNN), Zhao *et al.* [11] proposed a multiscale architecture to recognize multilingual characters. A new deep learning technique, directed acyclic graph - convolutional neural network (DAG-CNN) was


(a) Confusion matrix of doc_class_net classifier

Target class

Output class	Class 1	Class 2	Class 3	Class 4	Classification overall	Producer accuracy (precision)
Class 1	78	9	1	0	88	88.636 %
Class 2	11	81	3	0	95	85.263 %
Class 3	11	9	95	1	116	81.897 %
Class 4	1	1	1	99	102	97.059 %
Truth overall	101	100	100	100	401	
User accuracy (recall)	77.228 %	81 %	95 %	99 %		

Overall Accuracy (OA)	88.03 %
Kappa	0.84


(b) Confusion matrix of line_class_net classifier

Output class					
Cursive	51 22.3 %	31 13.5 %	0 0.0 %	0 0.0 %	62.2 % 37.8 %
Discrete	9 3.9%	28 12.2%	0 0.0%	0 0.0%	75.7 % 24.3 %
Printed	0 0.0 %	0 0.0 %	49 21.4 %	0 0.0 %	100 % 0.0 %
Semi printed	0 0.0 %	1 0.4 %	0 0.0 %	60 26.2 %	98.4 % 1.6 %
	85.0 % 15.0 %	46.7 % 53.3 %	100 % 0.0 %	100 % 0.0 %	82.1 % 17.9 %
	Cursive	Discrete	Printed	Semi printed	

Target class

Fig. 3 (a) The confusion matrix of the doc_class_net. The overall accuracy of 88.03 % bypassing a total of 401 test images into the model. Class 1, class 2, class 3 and class 4 correspond to semi-printed, handwritten cursive, handwritten discrete and printed respectively. The overall accuracy for semi-printed, handwritten cursive, handwritten discrete, and printed is 88.636 %, 85.263 %, 81.897 %, and 97.059 % respectively. (b) The confusion matrix of line_class_net giving the final accuracy of 82.1 %. The model was accurately able to identify the printed text images.The overall accuracy for cursive, discrete, printed and semi-printed is 62.2 %, 75.7 %, 100 % and 98.4 % respectively.

used by Bhagyasree *et al.* [12] for handwritten character recognition. Butala *et al.* [13] created a pen that can be used by a physician to write the prescription which will simultaneously transform the handwritten text into a printed form and display it online. The printed form will then be aligned, parsed, and formatted using formatting tools and thus the OCR techniques can be used in the field of medicine.

3 Methodology

Input given can be an image or a pdf document. If the input is given is a pdf document it is converted to images and then one by one these images are sent to a classifier that segregates it into printed, handwritten discrete, handwritten cursive, and semi printed as shown in Figure 1. The dataset for the handwritten images used was the IAM dataset, whereas, for the printed and the semi printed, the dataset was created by ourselves. After classification, if the image is printed it is passed to Pytesseract which is a pre-trained model for recognizing printed text. If the image is handwritten cursive or handwritten discrete then it is fed to a horizontal splitter that generates line-wise images. These images classified as handwritten discrete or handwritten cursive are predicted using a CL-9 model which is trained separately on discrete and cursive. If an image is classified as semi printed, it is passed to a horizontal line splitter and then the line-wise images are passed to the line_class_net classifier. The line_class_net classifier further segregates the line image into printed, semi-printed, handwritten discrete and handwritten cursive. If the output is printed, handwritten cursive, or handwritten discrete, we send the image to the respective model and if the output is semi printed, it is passed through a vertical splitter that splits the line image into two halves vertically. These two split images are fed again to the line classifier and the process is repeated until there's no semi printed image.

3.1 doc_class_net(Full size image classifier)

The dataset contained folders of printed text, handwritten cursive text, handwritten discrete text, and the semi-printed text was passed into the model. The model (Figure 2 (b)) was able to achieve almost full accuracy on the training dataset. The proposed model consisted of two convolutional layers followed by ReLU activation layers and then Normalisation and Max pooling layers. Initially, input images of size (227X227) were passed into the model which goes through the first convolution layer, then through the ReLU activation layer. The normalization layer ensures cross channel normalization with 5 channels per element. After going through the Max pooling layer it was passed to the second convolutional layer. It was further passed through similar layers and was then outputted by a fully connected layer after a dropout of 0.5. The softmax activation function was applied and the output was obtained. If the input

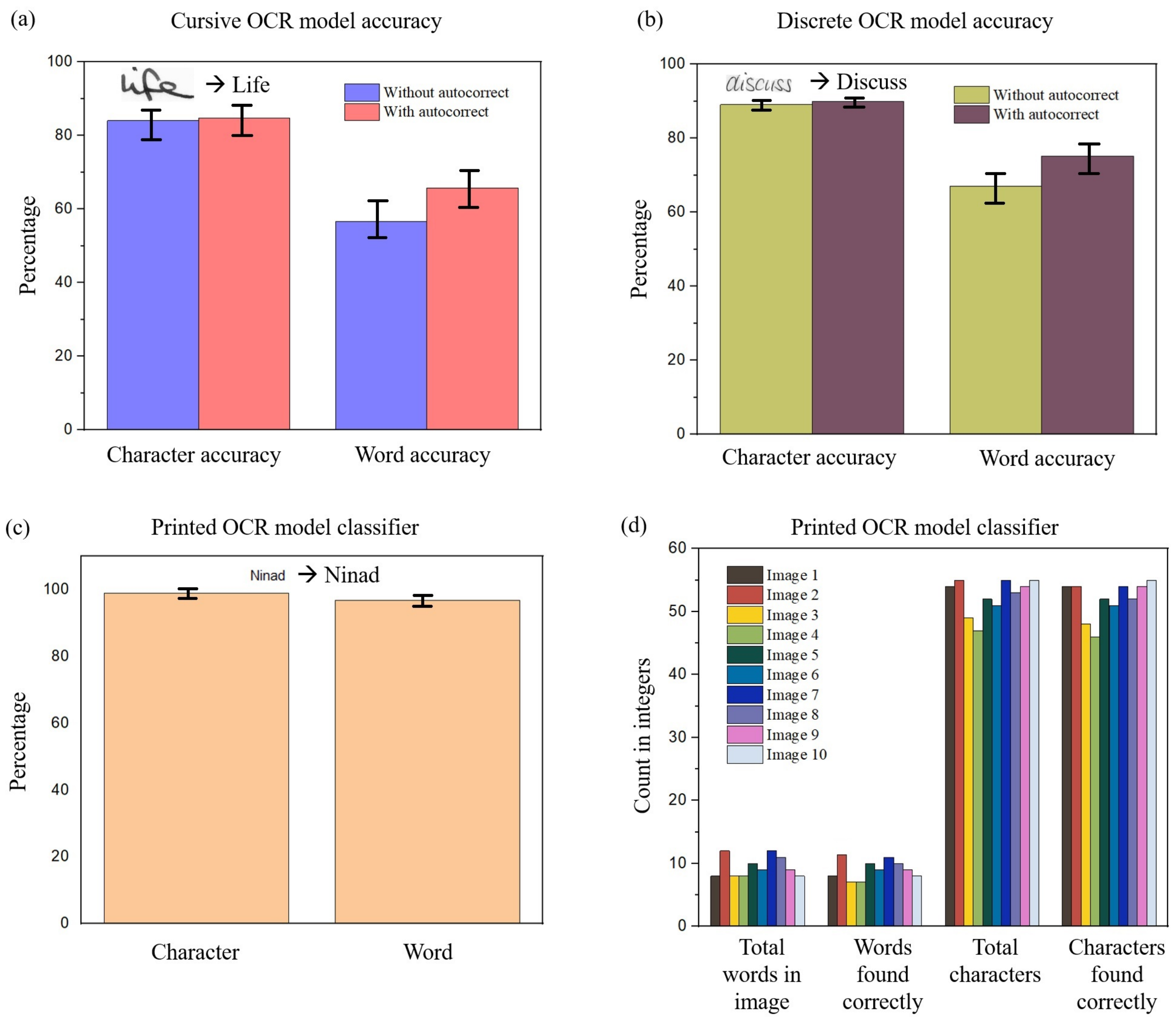


Fig. 4 (a) Histogram representing the CL-9 cursive model accuracy. A standard error of 5 % has been considered for the accuracy percentage. The blue bar indicates accuracy percentage without autocorrect and the red bar represents percentage accuracy with autocorrect.Here in the histogram we have shown snippet of the line given to the OCR for handwritten cursive which was "life" and the output was also shown as "life" as predicted by the OCR model for handwritten cursive. The graph shows accuracy for character level as well as word level. (b) Histogram representing the CL-9 discrete model accuracy. A standard error of 5 % has been considered for the accuracy percentage. The green bar indicates accuracy percentage without autocorrect and the purple bar represents percentage accuracy with autocorrect. The graph shows accuracy for a character as well as a word. Here in the histogram we have shown a snippet of the line given to the OCR for handwritten discrete was "discuss" and the output was also shown as "discuss" as predicted by the OCR model for handwritten discrete.(c) Histogram of printed OCR model for characters level and word level. A standard error of 5 % has been considered for the accuracy percentage. Here in the histogram we have shown a snippet of the line given to the OCR for printed text was "Ninad" and the output was also shown as "Ninad" as predicted by the Pytesseract. (d) A detailed accuracy histogram of a printed OCR model. 10 input lines were used as an input and named as image 1 to image 10. This shows how many total words were present and how many words were correctly found. Also, the count of total characters and the characters correctly found were represented.

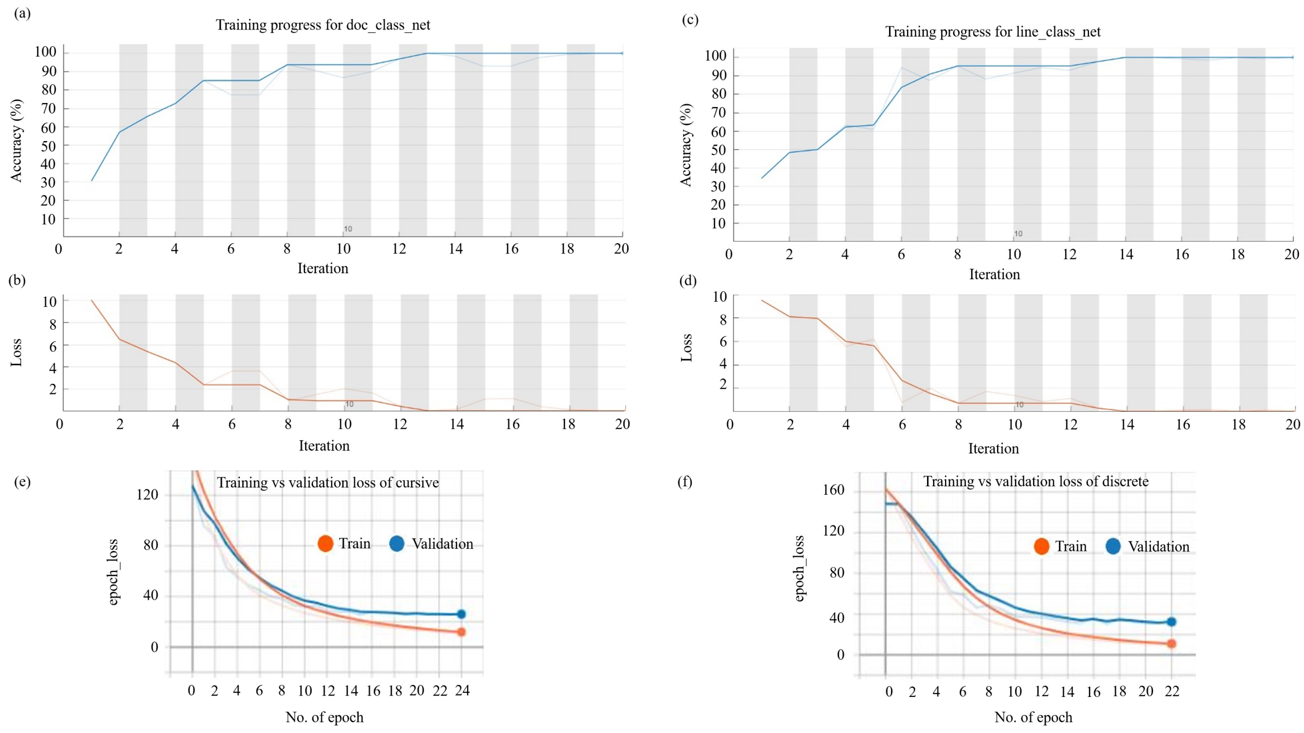
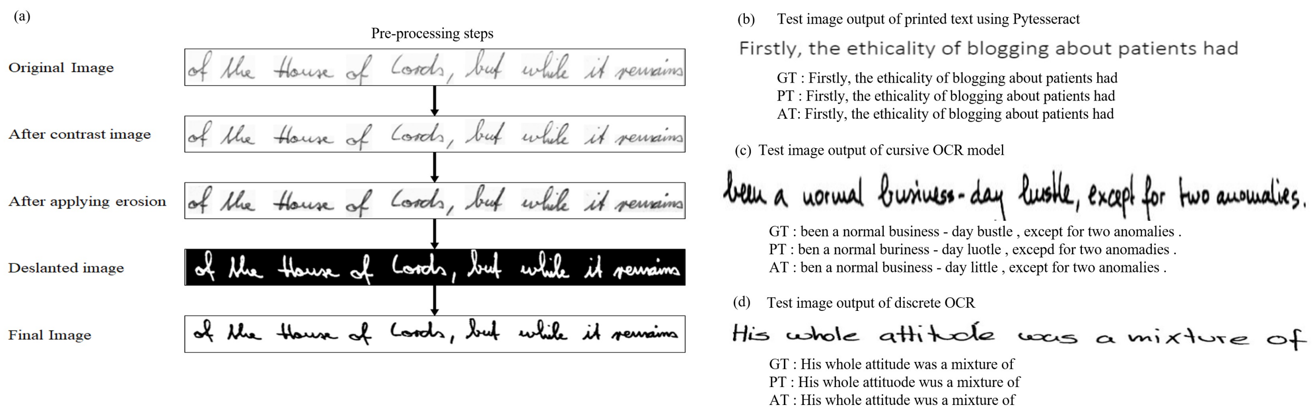


Fig. 5 (a) Accuracy graph for doc_class_net. After the 5th epoch, the accuracy reached up to 80 %, whereas after the 8th epoch it crossed 90 % accuracy, whereas, after the 13th epoch, the saturation level was reached. (b) Accuracy graph for the line_class_net. After the 5th epoch, the slope of the accuracy increases drastically. After the 8th epoch, the saturation level was reached. (c) Loss graph for doc_class_net. Till the 5th epoch, the loss decreased drastically, whereas between the 5th to 13th epoch, the loss decreased gradually. After the 12th epoch, it reached the saturation level. (d) Loss graph for the line_class_net. Till the 5th epoch, the loss decreased gradually, whereas between the 5th to 13th epoch, the loss decreased drastically. After the 12th epoch, it reached the saturation level. (e)Training versus validation loss of CL-9 cursive model. The blue line indicates the validation curve whereas the orange line indicates the training curve. The validation loss was lower than the training loss initially, then at the 6th epoch it became almost the same and at the end, the validation loss was more than training loss. (f) Training versus validation loss of CL-9 discrete model. The orange line indicates the training curve whereas the blue line indicates the validation curve. Till the 7th epoch both the losses were almost the same and then the difference between them gradually increases.


(a)
Pre-processing steps
Original Image
After contrast image
After applying erosion
Deslanted image
Final Image
of the House of Lords, but while it remains
(b) Test image output of printed text using Pytesseract
Firstly, the ethicality of blogging about patients had
(c) Test image output of cursive OCR model
been a normal business-day bustle, except for two anomalies.
(d) Test image output of discrete OCR
His whole attitude was a mixture of

GT : Firstly, the ethicality of blogging about patients had
PT : Firstly, the ethicality of blogging about patients had
AT: Firstly, the ethicality of blogging about patients had

GT : been a normal business - day bustle , except for two anomalies .
PT : ben a normal buriness - day luotle , excepd for two anomadies .
AT : ben a normal business - day little , except for two anomalies .

GT : His whole attitude was a mixture of
PT : His whole attituode wus a mixture of
AT : His whole attitude wus a mixture of

Fig. 6 (a) Preprocessing techniques applied to a sample text. The original images undergo various steps to obtain the final image. Contrast was applied first and then erosion was applied to the sample text. After the erosion was applied, the image was deslanted to obtain the final image. (b) Test image output of printed text using Pytesseract (c) Test image output of cursive OCR model. (d) Test image output of discrete OCR model. In (b), (c) and (d), GT stands for Ground Truth, PT stands for Predicted Text and AT stands for Autocorrect Text was fully handwritten or fully printed, the image is segmented into lines and given to the respective models. If the image is classified as semi-printed, the image goes through line segmentation and then to the line_class_net classification model.

3.2 Horizontal line splitter

In horizontal line splitter, we first converted the image to grayscale and then into a binary image. Thereafter, we dilated the image. We found the edges in the image after dilation and around that edge, we drew an approximate rectangle. If the two rectangles overlap or join, we merged them and with the help of the coordinates of that rectangle, we got the lines from the original image.

3.3 line_class_net classifier (Line-wise classifier)

The model (Figure 2 (c)) took line-wise input as images and classified it into either printed, handwritten discrete, handwritten cursive, or semi printed. Initially, input line-wise images of size (1000X100) were passed into the model which goes through the first convolution layer, then through the ReLU activation layer. The normalization layer ensures cross channel normalization with 5 channels per element. After going through the Pooling layer it was passed to the second convolutional layer. It was further passed through similar layers and is then outputted by a fully connected layer after a dropout of 0.5. The softmax activation function is applied and the output is obtained.

3.4 OCR for printed documents

Tesseract is an optical character recognition engine that is the most popular and qualitative OCR-library. OCR uses artificial intelligence for text search and its recognition of images. The Tesseract engine was trained on Long Short Term Memory(LSTM) layers which improved multistage processes like word-finding, line finding, and character classification. Pytesseract is a wrapper for Tesseract-OCR Engine. One can simply implement the Tesseract OCR Engine on python using its library - pytesseract. Pytesseract when implemented, first finds the lines in our input image, and then it applies baseline fitting. After these two steps, pytesseract implements fixed-pitch detection and chopping on characters and then finds the probability of possible characters using features of character recognition. Pytesseract uses Tesseract OCR algorithms and several LSTM layers along with efficient and accurate scene text (EAST) detection techniques to provide us with an output of extracted text present in the input image.The example of printed text with the true label and predicted text is shown in Figure 6 (b). However, Pytesseract didn't prove much effective for handwritten text images and hence we used our own CRNN model i.e. CL-9 model.

3.5 OCR for handwritten discrete and cursive

In the first step of preprocessing as shown in Figure 6 (a), the images are processed to increase the contrast. The second step involves morphological operation erosion to increase the thickness of the foreground object i.e. the text, as it is dark. The image was further converted into binary and then deslanted for better perception. The final preprocessed image was obtained after taking the binary and transpose. After this step, data augmentation can be applied. The proposed model (Table 1) consists of 7 convolutional layers followed by two Bidirectional LSTM layers. In each layer first, there is a Convo 2D layer followed by an Activation layer, Batch Normalisation, and MaxPooling. In an activation layer, PRelu was applied to introduce non-linearity. The input images were divided into batches of size 30 and the size of each input image is (1024, 128, 1). The learning rate of the model was set to 0.001. The feature maps help in the extraction of vectors. These were then fed into the first Bidirectional LSTM layer. This was followed by a dense layer and the second Bidirectional LSTM layer. The forward and backward outputs are combined at the end of each Bidirectional LSTM. This ensures the faster optimization of weights. The last layer is a dense layer with a total of 98 units(which is vocabulary size) with activation function softmax. The optimizer used while training was RMS optimizer and CTC beam search decoder was used for decoding the output to editable text. The output of the CRNN and the corresponding ground-truth text was fed to the CTC loss function to calculate the loss for the training samples. To recognize the text in test images, the images are passed through the trained CRNN model and to decode the output, CTC beam search decoding with a beamwidth of 10 was applied. In the post-processing phase, autocorrect was applied to improve the accuracy. The examples of the handwritten cursive and handwritten discrete images with the true label as well as both predicted text and autocorrected text are shown in Figure 6 (c), Figure 6 (d) respectively.

Layer		Feature map	Size	Kernel size	Stride	Activation
Input	Image	1	(1024, 128,1)			
1	Conv2D	16	(1024, 128, 16)	(3, 3)	(1, 1)	prelu
	Maxpooling2D		(512, 64, 16)	(2, 2)	(2, 2)	
2	Conv2D	32	(512, 64, 16)	(3, 3)	(1, 1)	prelu
	Maxpooling2D		(256, 32, 32)	(2, 2)	(2, 2)	
3	Conv2D	40	(256, 32, 40)	(3, 3)	(1, 1)	prelu
	Batch Normalization		(256, 32, 40)			
	Maxpooling2D		(128, 16, 40)	(2, 2)	(2, 2)	
4	Conv2D	48	(128, 16, 48)	(3, 3)	(1, 1)	prelu
5	Conv2D	56	(128, 16, 56)	(3, 3)	(1, 1)	prelu
	Maxpooling2D		(128, 8, 40)	(1, 2)	(1, 2)	
6	Conv2D	64	(128, 8, 64)	(3, 3)	(1, 1)	prelu
	Maxpooling2D		(128, 2, 64)	(1, 4)	(1, 4)	
	Batch Normalization		(128, 2, 64)			
7	Conv2D	128	(256, 2, 128)	(3, 3)	(1, 1)	prelu
	Maxpooling2D		(128, 1, 128)	(1, 2)	(1, 2)	prelu
	Reshape		(128, 128)			
8	Bidirectional LSTM	128	(128, 256)			
	Dropout		(128, 256)			
	Dense	256	(128, 256)			
9	Bidirectional LSTM	128	(128, 256)			
	Dense	98	(128, 98)			Softmax

Table 1 Architecture for handwritten cursive and discrete OCR model (CL-9 model)

Method	Accuracy (%)
Wei *et al.* [14]	90.6
Singh *et al.* [15]	94.55
Ramanathan *et al.* [16]	97
Chaudhari *et al.* [17]	99
Our proposed method	91.26

Table 2 Comparison of methodologies reported in existing literature

4 Results

The doc_class_net classifier (Figure 3 (a)) showed an overall accuracy of 88.03 % bypassing a total of 401 test images into the model. Class 1 includes the semi-printed documents consisting of both printed and handwritten data. Out of 88 semi-printed data, 78 were correctly classified into semi-printed whereas 9 were classified as handwritten cursive and 1 as handwritten discrete. The accuracy of the classification of sem-printed documents was 88.636 %. Class 2 consists of handwritten cursive data. Out of 95 handwritten cursive data, 81 were correctly classified whereas 11 were wrongly classified as semi-printed and 3 as handwritten discrete giving an overall accuracy of 85.263 %. Class 3 classifier includes the handwritten discrete data. 95 out of 116 were correctly classified as handwritten discrete whereas 11 were classified as semi-printed, 9 as handwritten cursive, and 1 as fully printed. The overall accuracy achieved was 81.897 %. The class 4 classifier consisted of 102 fully printed data. 99 were correctly classified whereas 1 was wrongly classified as semi-printed, 1 as handwritten cursive, and 1 as handwritten discrete giving an accuracy of 97.059 %. The confusion matrix of line_class_net classifiers (Figure 3 (b)) gives the final accuracy of 82.1 %. The model was accurately able to identify the printed text images. From a total of 82 images for handwritten cursive data, 51 were correctly labeled as handwritten cursive whereas 31 were classified into handwritten discrete, giving an accuracy of 62.2 %. Out of 37 images for handwritten discrete data, 28 were correctly classified whereas 9 were wrongly classified into handwritten cursive. The accuracy for the same was 75.7 %. 49 out of 49 images were classified into printed giving an accuracy of 100 %. 60 out of 61 images were correctly classified into semi-printed whereas 1 was wrongly classified into handwritten discrete. The accuracy for the same was 98.4 %. Classification of the handwritten text images into discrete and cursive was tough, however, the model gave good accuracy but there is still scope for improvement. For a total of 772 handwritten cursive images (Figure 4 (a)), the accuracy of OCR for the character was 84.75 % and word accuracy was 65.74 % with autocorrect. Whereas without autocorrect, the accuracy for character classification was 84.04 % and word accuracy was 56.6 %. For a total of 708 handwritten discrete images (Figure 4 (b)), the accuracy of OCR character-wise was 90 % and word accuracy was 75.20

% with autocorrect. Whereas without autocorrect, the accuracy for character-wise was 89.14 % and word accuracy was 67 %. For the printed OCR (Figure 4 (c)) we got 94.79 % word accuracy and 99.04 % character accuracy without using autocorrect. As shown in Figure 4 (d) a total of 96 words were present in all the 10 images considered together. 91 words were correctly recognized, giving an accuracy for the word level OCR as 94.79 %. A total of 525 characters were present in all the 10 images, and 520 characters were correctly identified giving an accuracy of 99.04 % for the character level OCR. The accuracy achieved for images 1, 5, 6, 9, and 10 was 100 %. As shown in table 2, we have compared our methodology with the other methodologies reported in the literature. Most of the literature has used the character-wise classification for the input rather than the line-wise approach which reduced our time compared to other methodologies. We have firstly classified the input images into four different groups and then applied OCR on them. We achieved 91.26 % overall accuracy of printed, handwritten cursive and handwritten discrete for the OCR model at character level which was higher than the accuracy 90.6 % achieved by Wei *et al.* [14]. Chaudhari *et al.* [17] achieved the highest accuracy of 99 % for the printed english text through the soft computing techniques. Ramanathan *et al.* [16] used the Gabor filters and support vector machines and achieved an accuracy of 97 %. Singh *et al.* [15] achieved an accuracy of 94.66 %.

5 Discussions

The training was tried with 6 to 9 convolutional layers. Best results were obtained from 7 layers and hence, the model with 7 layers was finalized. After that three models were designed by changing the number of filters in each layer where the first model ranged from 16 filters to 128 filters, second one ranging from 32 filters to 256 filters and third one ranging from 64 to 256 filters. Out of these three models, the first and third models gave better and almost the same accuracy but the time taken by the third model was much more than the first model. So, the model with filters starting from 16 filters was finalized. The training was also performed by changing preprocessing techniques. Data was first passed by simple processing techniques such as binarization and deskew. Then the deslanting algorithm was applied which gave better results than simple preprocess techniques. Data augmentation was also applied but no considerable change was detected hence that part was removed. Images containing diagrams and graphics were not taken into consideration. The model was trained only on images containing text. No preprocessing technique was tried for the images with graphics.

6 Conclusion

We proposed a system that first classifies the input image into four classes and then applying optical character recognition to printed, handwritten cursive, and handwritten discrete. If the document is fully handwritten cursive or discrete, then it uses convolutional recurrent neural networks (CRNN). If printed, then it uses OCR of the tesseract to get an editable text. This feature becomes extremely useful when dealing with sequential data. With OCR processing tools, we can extract data from scanned documents and put them into databases, so that information is quickly recorded and retrieved when needed in the future. This will also prevent loss of data which is possible with the traditional method of storing paper records if the proper backup of the electronic databases is taken.

Acknowledgements Authors would like to thank all colleagues who helped us in completion of the work.

Compliance with Ethical Standards

Conflicts of interest

Authors A. Tiwari, D. Raikar, J. Chauhan, P. Harniya, S. Bagwe, V. Gada, V. Gupta, V. Mehta, V. Shah, U. Bheda, M. Warang and N. Mehendale, declare that they have no conflict of interest.

Involvement of human participant and animals

This article does not contain any studies with animals or Humans performed by any of the authors. All the necessary permissions were obtained from the Institute Ethical Committee and concerned authorities.

Information about informed consent

No informed consent was required as the studies do not involve any human participant.

Funding information

No funding was involved in the present work.

References

1. C. Patel, A. Patel, D. Patel, Optical character recognition by open source ocr tool tesseract: A case study, International Journal of Computer Applications **55**(10), 50 (2012)
2. L.R. Blando, J. Kanai, T.A. Nartker, in *Proceedings of 3rd International Conference on Document Analysis and Recognition*, vol. 1 (IEEE, 1995), vol. 1, pp. 319–322
3. A. Graves, S. Fernández, F. Gomez, J. Schmidhuber, in *Proceedings of the 23rd international conference on Machine learning* (2006), pp. 369–376
4. D. You, S. Antani, D. Demner-Fushman, V. Govindaraju, G.R. Thoma, in *2011 International Conference on Document Analysis and Recognition* (IEEE, 2011), pp. 967–971
5. M. Mariappan, V. Ramu, T. Ganesan, B. Khoo, K. Vellian, in *Proceedings from the International Conference on Biomedical Engineering and Technology (IPCBEE)*, vol. 11 (2011), vol. 11, pp. 70–74
6. S. Kumar, N.S.A. Deep, K.G.M.R. Ghosh, Offline handwriting character recognition (for use of medical purpose) using neural network, International Journal Of Engineering And Computer Science **5**(10) (2016)
7. D.A. Huang, L. Fei-Fei, J.C. Niebles, in *European Conference on Computer Vision* (Springer, 2016), pp. 137–153
8. A. Mithal, P. Kumaraguru, Optical character recognition tool (2017)
9. W.H. Weng, K.B. Wagholikar, A.T. McCray, P. Szolovits, H.C. Chueh, Medical subdomain classification of clinical notes using a machine learning-based natural language processing approach, BMC medical informatics and decision making **17**(1), 1 (2017)
10. H. Scheidl, S. Fiel, R. Sablatnig, in *2018 16th International Conference on Frontiers in Handwriting Recognition (ICFHR)* (IEEE, 2018), pp. 253–258
11. Y. Zhao, W. Xue, Q. Li, in *2018 IEEE Fourth International Conference on Multimedia Big Data (BigMM)* (IEEE, 2018), pp. 1–5
12. P. Bhagyasree, A. James, C. Saravanan, in *2019 1st International Conference on Innovations in Information and Communication Technology (ICIICT)* (IEEE, 2019), pp. 1–4
13. S. Butala, A. Lad, H. Chheda, M. Bhat, A. Nimkar, in *2020 International Conference on Emerging Trends in Information Technology and Engineering (ic-ETITE)* (IEEE, 2020), pp. 1–7
14. T.C. Wei, U. Sheikh, A.A.H. Ab Rahman, in *2018 IEEE 14th International Colloquium on Signal Processing & Its Applications (CSPA)* (IEEE, 2018), pp. 245–249
15. D. Singh, M.A. Khan, A. Bansal, N. Bansal, in *2015 Communication, Control and Intelligent Systems (CCIS)* (IEEE, 2015), pp. 167–171
16. R. Ramanathan, S. Ponmathavan, N. Valliappan, L. Thaneshwaran, A.S. Nair, K. Soman, in *2009 International Conference on Advances in Computing, Control, and Telecommunication Technologies* (IEEE, 2009), pp. 610–612
17. A. Chaudhuri, K. Mandaviya, P. Badelia, S.K. Ghosh, in *Optical Character Recognition Systems for Different Languages with Soft Computing* (Springer, 2017), pp. 85–107