Grenze International Journal of Engineering and Technology, Jan Issue

Comparative Study on Handwritten Character Recognition Tools

Arthik Bhandary[1] and Anala M R[2]
[1]Computer Science Department, RV College of Engineering, Bengaluru, India
Email: arthikrb@gmail.com
[2] Information Science Department, RV College of Engineering, Bengaluru, India
Email: analamr@rvce.edu.in

***Abstract*—Recognition of handwritten characters has been undergoing a lot of research and development nowadays, since it could potentially transform various field by allowing industries and fields to undergo complete digitization among other things. But recognizing handwritten characters has been observed to be quite difficult due to fact that writing styles differ from person to person, leading to different 'fonts' for different persons. This problem exists in every language. Recently there have been research advances mainly in machine learning and deep learning which show great promise for the future of this field. In this paper, various approaches towards implementing character recognition have been explored.**

***Index Terms*— Handwritten character recognition, pre-processing, feature extraction and character classification.**

I. INTRODUCTION

Nowadays, huge amounts of information are being stored, processed and transferred in digital form. Information and data stored in the digital form has lots of advantages over the conventional paper with almost few to no disadvantages. Digitalization of data would save lots of time for institutions, companies and individuals, it would lead to reduction in costs, in the form of manpower, paper costs, storage space, maintenance and labor. But still a large part of the data and information are still written in papers and other forms and are finally stored as such. Even though this ultimately results in increased cost and time for the people involved. An important reason for this is the difficulty in converting information and data held in physical forms into digital forms. This process is generally associated with people typing and feeding these data to a computer which is tiresome, labor intensive and quite expensive.

To overcome this problem a new technology and field was born, Optical Character Reading (OCR). OCR deals with conversion of information stored in physical forms such as papers into electronic data by processing scanned documents or images. OCR is generally associated with converting printed documents. But most of the information and data stored in physical forms contain handwritten texts in them which cannot be processed and recognized by ordinary OCRs. This could be attributed to the fact that different people have different writing styles, which become even more complicated in the case for multi-lingual OCRs. Thus, Handwritten/Intelligent Character Recognition (ICR) [1] was born, which specializes in recognizing and identifying handwritten and printed documents with various types of fonts

Handwriting recognition system can be roughly categorized into two types, that is, online and offline character

Grenze ID: 01.GIJET.8.1.4_1

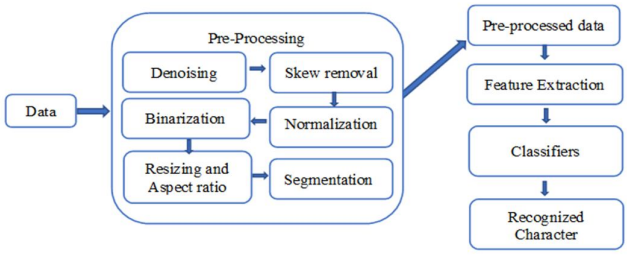

Figure 1: General Steps in a Character Recognition Algorithm

recognition [2]. Online handwriting recognition deals with recognition of text while it is being written. Off-line character recognition involves recognition of characters present in images. This is a static approach towards recognition. Off-line handwriting recognition is more challenging than on-line recognition due to quality of image and absence of extra parameters available in online systems. But offline recognition has more practical use since it does not require specialize equipment except for processing. It also works on documents and papers unlike online recognition.

Modern ICRs are quite developed for popular and global languages, like English, being able to provide accuracy as high as 97% [3]. But for regional languages like Kannada, Telugu, Hindi etc., ICRs have still not been developed extensively. Such types of multi-lingual ICRs are really challenging and difficult to develop due to the incorporation of multiple scripts for the different languages to the program. The characters in the different scripts could be similar too which would result in wrong identification and decreased accuracy rates.

Applications of character recognitions can be seen in healthcare and education too. They can also support blind and visually impaired users, making their lives easier by relaying information in their surroundings, like papers, signs or posters etc., if necessary. Thus, such technology can act as 'reading machine' for the visually impaired. This paper focuses on exploring tools available for off-line handwriting recognition involving new and novel methods of developing and implementing ICRs for the purpose of recognizing handwritten characters and converting them into digital data.

II. Literature Review

There has been lots of research and developments in the area of character recognition. Generally, every approach can be broken down into three steps, namely, pre-processing, feature extraction and recognition. Fig. 1 represents the general steps followed in character recognition approach.

Recognition system based on the segmentation approach developed for handwritten Hindi text was focused in [4]. Initially text is segmented into lines and then into words. Upper and lower modifiers are obtained from the words. Consonants, half characters, and joint characters are recognized and categorized. For the purpose of recognition and identification of characters, rule-based classifier was implemented. The average rate of recognition for characters which had been correctly segmented was found to be 88.75%. The errors in this method were mostly due to fault in segmentation, which was usually because of the over segmentation of lower modifier or under segmentation of upper modifier.

Application of deep learning approaches in intelligent character recognition, has shown great promise and breakthroughs with the usage of lexicon-based architectures and recurrent neural networks [1]. Along with adopting Convolutional Neural Network (CNN) architecture, a probabilistic character rate has been introduced. For common words, the lexicon is integrated into the process. A symbol prediction fully convolution network forms the core of this approach. The performance depends on the alignment of its output to ground truth symbols.

Ref. [3] proposes a method where wavelet coefficients and convolutional networks were combined for extracting features. Vertical, horizontal and diagonal wavelet coefficients are given as first, second and third input respectively. By using these coefficients as inputs, it can be assured to a level that only relevant information is supplied to the network as features. Accuracy of Recognition of characters was found to be 96.58% in addition to reduced training time

Text localization is done to identify bounding boxes in every line of text [5]. An SSD: Single Shot multibox Detector can be implemented for the purpose of predicting bounding boxes relative to anchor points and to predict the probability, that these boxes contain any words. A multi-scale CNN is implemented for extracting features which are then given as input to a bidirectional Long Short-Term Memory (LSTM).

In 2018 [6], a method was proposed that made use of Histogram of Oriented Gradients (HOG) features present in the input image of individual characters as a basis to classify characters. Use of image partition techniques are done to generate feature vector. For the purpose of segmentation of lines in the given images, horizontal image projection is utilized whereas vertical intensity histogram projection is utilized for segmentation of each character in every line. A two-level classification scheme has been proposed which serves the purpose of increasing the accuracy of the recognition of characters.
Principal Component Analysis (PCA) [7] has been utilized to determine and select prominent features, which are then used to identify the handwritten characters. PCA involves implementation of linear transformations to map data from higher dimensional space to lower dimensional space. In this method, the low dimensional space is determined with the help of Eigen vectors of the covariance matrix. Eigen vectors having the largest eigen value is considered to be the most significant and then the one with the second largest is considered as the second most significant.

III. Pre-processing

One of the most vital and the first aspect of feature extraction is pre-processing. Pre-processing refers to several stages done before the actual classification, such as skew detection and correction, binarization, removal of noise, thinning etc. Pre-processing can help in the feature extraction and character detection by trying to removing irrelevant information from the input images.
Skew Detection and Removal: This usually happens when a text or the document was not scanned properly and has been skewed by angle. By removing the skew, the recognition can be improved. Depending on the data set, the input image might have to be rotated by an angle of 90,180 or 270 degrees. This must be corrected too.
Binarization and Normalisation: Binarization refers to the process of converting the input document from colour to grayscale. By doing binarization, a lot of extra information is removed such as colour which are usually not necessary for the recognizing characters. This is because the colour of the text does not affect the meaning of the text and would be irrelevant for character recognition. It is observed that recognizing text in binary images is easier too. This leads to increased accuracy and effectiveness in character recognition. Many of the recognition algorithms work exclusively on binary images, since working on colour images tends to make the algorithm complicated without any useful gain. The pixels hold value within a range 0 to 255. Normalization refers to adjusting values to change range. Usually, it is normalized to a range 0-1 or -1 to 1, etc. This tends to help in speeding up the convergence of the model. In the case of neural network, it also prevents the network from going out of control.
Denoising: Noise refers to random variation in the features. Usually, the input image contains noise which tends to affect the accuracy of the algorithm. Removing noises can be tricky and complicated without affecting the relevant information.
Thinning: In this process, the width of the lines in the text is reduced, usually to one pixel length. This is done to obtain only the skeleton of the text, for the purpose of character recognition. There are various thinning algorithms like Guo-Hall algorithm, Enhanced Parallel thinning algorithm etc.
Resizing and aspect ratio: Usually the algorithms used for recognition, may require the same size or aspect ratio for every input data, but usually this is not the case. Thus, the input images can be resized or padded to satisfy the required input size.
Segmentation: Segmentation is to separate the images to units that include paragraphs, line and even further into words and letters. This process can be done by considering the spaces between letters and words but this proves difficult in cursive or scripts with no spaces between each letter in a word, i.e., the letters are continuous. Segmentation forms a vital part of character recognition. Even though it has been included in the pre-processing stage. segmentation can be considered another stage in itself. Since it is usually the case that, the data available isn't separated into characters and are images or scans of documents. Thus, it becomes vital that segmentation is done properly and properly segmented characters are sent to the next stage. Segmentation can speed up training, and also helps to avoid incorrect classification.

IV. Feature Extraction Techniques

Feature extraction is another vital aspect of character recognition algorithm which can result in the failure of the algorithm if done incorrectly. The pixel values of the image itself can be used as a feature, while properties like loops, lines, intersection point, branches, joints, curve, strokes and end points can also be calculated and used as a feature. Some promising feature extraction techniques are as follows

A. Cross-Corner Feature Extraction

In this approach [8], the pre-processed binary image is split into n zones, which allows for extraction of local features instead of global ones. From each of the zones, left and right slanted lines, i.e., diagonals from the corners of the zones are identified and counted. In each zone, the number of left and right diagonal become two features of respective zones. Dividing into zones and extracting the diagonal lines in each zone, makes it easier to differentiate similar characters like 'X' and 'Y' compared to other diagonal methods. The features obtained are then combined into one feature vector and used for the purpose of classification.

B. F-ratio Weighted Feature Extraction

In many scripts, there are characters which are very similar to each other in appearance and hence most of their features match each other. In such cases, F-ratio [9] can be used to improve recognition of characters with similar patterns. The features are given a weightage based on this ratio, which helps in distinguishing them. F-ratio [9] is given by,

$$F=s^2_{bi} / s^2_{ai} \quad (1)$$

where s^2_{ai} is the between-class variance and s^2_{bi} is the in-class variance that are given by,

$$s^2_{wi} = \sum_{l=1}^{L} P(w_l) E\{(x_i - m_{li})^2 | w_l\}$$

$$s^2_{bi} = \sum_{l=1}^{L} P(w_l)((m_{li} - m_{0i})^2)$$

$$m_{li} = E\{x_i | w_l\}$$

$$m_{oi} = E(x_i) = \sum_{l=1}^{L} P(w_1)\, m_{li}$$

Where L is the class size, and P represents priori probability of a class

C. Discreet Cosine Transform (DCT)

This method [10] is used to transform the input data into their elementary frequency components. The coefficients obtained are stored in an array, with those with higher value placed near the top left. The general equation for 2D DCT for an N×M image is given by,

$$F(u,v) = \left(\frac{2}{N}\right)^{1/2} \left(\frac{2}{M}\right)^{1/2} \sum_{i=0}^{N-1} \sum_{j=0}^{M-1} \alpha(i)\alpha(j) \cos\left[\frac{\pi u}{2N}(2i+1)\right] \cos\left[\frac{\pi v}{2M}(2j+1)\right] . f(i,j) \quad (2)$$

$$\text{where, } \alpha(x) = \begin{cases} \frac{1}{\sqrt{2}} & for\ x = 0 \\ 1 & otherwise \end{cases}$$

In Ref. [11], four variances of coefficient such as Upper Left Corner (ULC), zigzag coefficients, Block based ULC and Block Based zigzag coefficients were implemented. For ULC coefficients, DLC is applied to the image, and then some N significant coefficients in the upper left corner are collected. N is chosen by experimentation and greater its value, higher is the information retained. In DCT zigzag coefficients, the data are extracted in a zigzag manner from top left to bottom right. In Block Based DCT, the image has to be divided into blocks, and DCT is applied to each of these blocks, from which the coefficients can be obtained.

D. Chain Code

In this technique, a sequence of code is used to represent the boundary of the character. The code contains the direction of succeeding points of the boundary. There are two approaches to chain code, 4 and 8 neighbourhood approaches. In the 8-neighborhood chain code, if the 8 directions are represented by numbers 0 to 7, the letter 'l' might be represented as '44444', if the starting point is the top of the letter, assuming 4 represents the down direction.

E. Wavelet Transform

This can be used for dividing data into different frequency components. Unlike basic Fourier Transform, which is localized in frequency, whereas wavelet transform is localized in frequency as well as time. The general form of wavelet transfer is given by (3),

$$f(a,b) = \int_{-\infty}^{\infty} \varphi^*_{(a,b)}(x)dx \quad (3)$$

In (3), the function φ, could be any function that follows some rules. The * represents the complex conjugate.
Wavelet Transform can be categorized into Continuous Wavelet Transform (CWT) and Discrete Wavelet Transform (DWT). DWT deals with orthogonal wavelets obtained while CWT makes usage of non-orthogonal wavelets. In [12], they have implemented DWT of the Symlet family. DWT returns four types of coefficients, approximation coefficients which are coarse scaled coefficients, and the remaining coefficients, diagonal, vertical and horizontal coefficients are fine scale coefficients. These are then provided to inverse DWT which returns a feature set. Discreet wavelet transfer provides various advantages like easy implementation, reduced computation time and decreasing the amount of resources needed.

F. Curvelet Transform

Curvelet Transform was proposed to overcome the shortcomings that are present in traditional multiscale representations. It is an extension of wavelet transform with higher dimensionality, designed to represent images at various angles and scales. It was observed that curvelet can be sufficiently represented by few coefficients and in a non-adaptive way, with increased directional sensitivity. Curvelets via Unequally Spaced Fast Fourier Transform (USFFT) and curvelets via wrapping transform introduced in 2005 [13] compared to the existing methods at the time were relatively simpler, efficient and faster. Irregular sampling of the Fourier coefficients of the input image were done to obtain the required coefficients in curvelets via USFFT. Whereas wrapping transform makes use of a series of translations along with a wrapping of specifically chosen Fourier samples. Wrapping transform was found to be faster, less redundant and simpler.
Curvelets via wrapping transform have been implemented by Pasha and Padma [14]. The algorithm used consists of two parts wrapping and inverse wrapping digital curvelet transform. The large number of coefficients produced in curvelet transform were reduced by making use of PCA. The usage of curvelets resulted in robust results and an increase in accuracy compared to approaches using other extraction methods.
There are still many other feature extractions other than the ones mentioned here such as multi-directional features, gradient features, Histogram of Oriented Gradients (HOG) based features, Kirsch Directional Edges, Neighbourhood Pixels Weights etc. But there doesn't any exist any feature extraction technique that can extract all the relevant features that can be used to recognise a character. Thus, usually, a hybrid of various feature extraction methods is used, so as to increase the accuracy of the recognition and ensure that relevant features can be extracted. In some approaches, feature extraction is integrated along with the classifier. In such cases, neural networks are usually used to extract features. The use of neural networks to extract features have gained attention in the recent years, especially with the advent of convolutional networks, since they can outperform traditional feature extractions in most cases.

V. CHARACTER RECOGNITION TECHNIQUES

The features extracted from the pre-processed image are given as an input for a recognition algorithm which has the capacity to distinguish and recognize the characters. These recognition algorithms should be initially provided with a sample dataset to train on. There are various kinds of recognition method and some of them have been discussed here. The recognition rate of an approach depends on the effectiveness of the feature extracted and also on proper implementation of these classifier algorithms.

A. Support Vector Machines (SVM)

SVM is a learning algorithm generally used for regression analysis and classification by providing a hyperplane separating the classes. The optimal hyperplane aims to maximize distance between two classes. SVM can make use of multiple hyperplanes to achieve classification into multiple classes. The original approach could only linearly separate the data, but by mapping the original dimensional space into a much higher dimensional space and defining them in terms of a kernel function. The separation in this higher dimensional space generally would be easier.
Ref. [15] made an implementation of Chain Code features along with SVM which was trained on a NSIT dataset. The SVM made use of a radial basis function as the kernel function, in order to allow non-linear separation of classes. This resulted in considerable increase in result accuracy compared to the previous approach. For lower- and upper-case English characters an accuracy of 86% and 88% was obtained, although when the model was trained and tested on a combined dataset, the results reduced to 73.4%. Ref. [16] made use of Moore's Neighbourhood Tracing to extract the boundary of the text. This considers a cluster of 'dark' pixels and starts from any one of these pixels and travel to neighbouring dark pixels, selected in a clockwise manner, and continues until it cannot move further or encounters the first pixel. Chain code is applied to this extracted

boundary. This is utilized by a linear SVM for the purpose of classification. The results were greatly affected by the type of inputs and failed to separate similar characters like J and I. The results of linear SVM were unsatisfactory, since they fail to capture higher dimensional features properly, but the use of non-linear SVM could possibly lead to better results.

C. K Nearest Neighbour (K-NN)

K Nearest Neighbour Classifier is a classifier algorithm, which classifies cases by considering the most common class amongst its nearest neighbours. K refers to the number of closest neighbours that must considered for classification and can be chosen through the experimentation.

Ref. [14] implemented a new approach by making use of K-NN Classifier along with PCA and Wrapping based Curvelet Transform. In this approach, curvelets were translated at every scale and angle with the help of a spatial grid. Even though wrapping based curvelet transform were faster and simpler than other approaches, it resulted in a large number of features. Working directly on this feature set, would increase training and prediction time considerably. Hence, PCA was used to reduce the dimensionality of the feature set. A K value of 3 was found to show good results and an accuracy of around 90.0% was achieved.

Ref. [17] made use of eccentrity and metric parameter as feature for the classifier. Eccentrity determines closeness of the object to a circle. Metric parameter can be defined as area by perimeter of the object. By training the K-NN classifier, these features can be grouped and used to identify the characters. A dataset containing 520 images was used and an accuracy of around 85% was achieved.

K-NN were found to show promising result but also suffer a few shortcomings. They aren't suited to very large datasets. Noise and outliers in the dataset have adverse impact on the classifier. Usually, feature scaling needed to be done. But in character recognition, especially for larger datasets, it is quite hard to avoid outliers and noise. Since K-NN is a lazy learner, it has a longer prediction time. These are not desirable in character recognition, where comparatively longer training time are acceptable, but predictions should be fast.

D. AdaBoost Classifier

This is a meta-estimator algorithm [18] having a classifier of the form

$$F(x) = sign\left(\sum_{t=1}^{T} \theta_t f_t(x)\right) \quad (4)$$

Here, f_t represents a weak learner which predicts the class of an input x and θ_t is the weight assigned to the classifier. The output of the classifier is calculated through the combination of the weighted sums of outputs of other learning algorithms. This classifier focuses on incorrectly classified cases on every subsequent iteration by increasing its weight. In this algorithm, a classifier is fit into the original dataset and subsequently fits copies of classifier on it, by increasing the weights of instances that had incorrect predictions. Thus, multiple learners come together to form a strong learning algorithm. Finally, the model is obtained by the weighted sum of all the learners. AdaBoost can also help in removing unwanted features.

Ref. [19] implemented AdaBoost Classifier for character recognition. On training the classifier on 3830 samples and testing on 78083 samples it achieved an accuracy of 98.6%. Whereas K-NN, trained and tested on the same dataset, only managed an accuracy of around 96.9%. Although AdaBoost needs more training time and is difficult to understand, it provides a better prediction accuracy and is less affected by noise. It also showed improvement in prediction speed and did not require feature scaling.

E. Artificial Neural Network (ANN)

ANNs have proved effective in character recognition since they try to mimic the working of human brain and are able to learn the weights necessary to recognising the characters. They have proven incredibly useful for solving various computational problems. Ref. [20] implemented 2D Complex Wavelet Transform with ANN. The 2D-CWT coefficients and geometrical features have been used for classifying characters. The use of hybrid of these two features made the ANN more stable, allowed for faster convergence and achieved high accuracy (99%), but it was trained on the MNIST datasets and was used only for numeral recognition. Further, Ref. [12] made use of ANN for the purpose of recognition of Kannada characters and numerals. For feature extraction, Discrete Wavelet Transform (DWT) was used, which resulted in a decrease in computation time and resources compared to other wavelet transforms. In order to obtain structural features, the image is divided into four quadrants and finally, quadrant density, correlation, aspect ratio and corner detection constitute structural features. By using almost 4350 training samples, they have managed to achieve an accuracy of 91.00% in character recognition. This approach was able to achieve higher accuracy compared to other approaches like K-NN with Hybrid Technique (87.33%) and K-NN with Quadtree (85.43%) that used similar datasets.

The ANNs were able to perform consistently better than other approaches, especially for character recognition whereas there were only slight improvements in numeral recognition. This could be attributed to the usage of structural features after dividing the image into quadrants, and thus better extraction of local features, which would lead to better separation and recognition similar characters and also because ANNs are better at classifying inputs into large number of character classes compared to the previous approaches.

F. Convolutional Neural Networks (CNN)

CNNs are specialised neural networks that implement a mathematical linear operation known as convolution, in some of their layers. CNNs include a convolutional, pooling and fully connected layers. Convolutional layers make use of a convolutional kernel or filter to convolve the given data and pass its result to the next layer and pooling layers combine cluster of output from the neurons into one for the next layer by applying some statistic operation such as max, min and average etc. The raw input image can also be used as an input for the CNN, bypassing the need for feature extraction but since raw images also contain redundant and unnecessary data would necessitate for far deeper CNN.

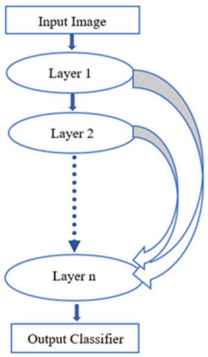

Figure 2: DAG-CNN

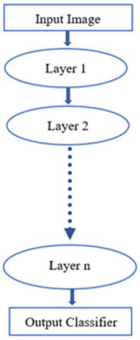

Figure 3: Basic CNN

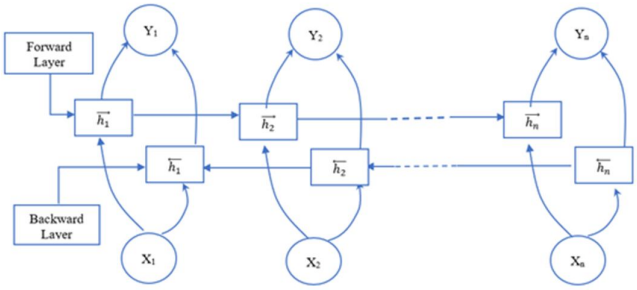

Figure 4: Bi-LSTM Network

Ref. [3] proposed a combination of wavelet coefficients and CNN where horizontal and vertical wavelet coefficients with diagonal features were provided to separate convolution layers. Relu activation function was used and the results were combined in the denser layer. This lead to decrease in training time compared to a basic CNN, with faster convergence and higher recognition rate. This has led to an increase in accuracy in character recognition, achieving an accuracy of 96.58%. Also, in 2020 [21] a combination of CNN and Error Correcting Output Code (ECOC) Classifier was introduced. While CNN is used for extracting features, ECOC has been used to classify the characters. They have implemented four CNN architectures, namely, LeNet of Type 1 and Type 2, AlexNet and Zenet. The network was trained on the NIST dataset, and the use of ECOC lead to increase in recognition rates, in every architecture except type 1 LeNet. d

Deep Learning techniques like CNN can classify and recognize characters with good accuracy. But basic CNN still might find it difficult to distinguish similar characters. In fact, this is the case with almost every approach discussed previously. This was addressed in feature extraction by making use of local features, but it cannot provide features that separate similar characters consistently. Also, in CNNs, every convolution layer is connected to the next convolution layer and only the last convolution can be connected to the output layer, this could lead to losses in some data that were present in the earlier layers, the deeper the network becomes, due to vanishing of gradients. Some seemingly insignificant features might be lost which would be vital to separate similar characters like ‘u’ and ‘v’ etc., especially in the case of handwritten characters.

To overcome this, a framework [22] was proposed that makes use of Directed Acrylic Graph-CNN. DAG-CNN avoids this problem by skipping connections, i.e., by making it possible for any layer to be connected to the any later layers, as shown in Fig. 2. In case of basic CNNs, as in Fig. 3, every layer is just linked to the next layer. This makes features of all levels in the CNN readily available, and reduces data loss. This results in the network being connected as a directed acrylic graph, but which would also lead to a very large number of features. Thus, only optimal layers must be selected, which was done by the utilization of a greedy forward selection strategy. The proposed framework was robust and less affected by noise. But there is also an increase in the amount of hyperparameters that would need to configured.

Another approach to separate similar characters in the absence of any distinguishing characters is to consider the surrounding characters of a character while predicting it. The surrounding characters could be incorporated as a

feature which effect the character classification process. When the classifier has high probabilities for multiple characters, the surrounding character would be better equipped to handle this ambiguity. So, a framework [5] was proposed based on CNN-biLSTM (Long Short-Term Memory) network. A multi-scale CNN had been used to extract features from the image and provide it to the bi-LSTM network. The CNN generates image features which are spatially aligned to the given image. The slicing of image features along the direction of the text is done for the purpose of generating a fixed number of timesteps which is then input to a LSTM. Connectionist Temporal Classification (CTC) loss was used as basis for training the network. The bidirectional LSTM allowed for classification of a character based on the previous and next characters. This allowed for better recognition of similar characters even with absence of any distinguishing features, through prediction by considering the previous and next characters.

As shown in the Fig. 4, activation values from the next and previous data help to determine the classification of the current output. The framework was trained on the IAM dataset containing 1539 documents, and was able to achieve similar accuracy to other approaches, while decreasing the memory and time required. Further research and development in this approach, could lead to promising results. Bi-LSTM could possibly bypass mistakes and predict the corrected character based on the surrounding characters, which would be invaluable in character recognition technology. For example, if a character is similar to 'n' and 'h', and if the previous and next characters are 't' and 'e', then it can be predicted with good accuracy to be 'h'.

TABLE I. COMPARISON OF VARIOUS CHARACTER RECOGNITION METHODS

Ref	*Feature Extraction Algorithm*	*Classifier*	*Datasets*	*Recognition Rate*	*Comments*
[15]	Chain Code	Radial Basis SVM	NSIT	73.4% English	Affected by noise. Not effective on large datasets
[14]	Wrapping Based Curvelets and PCA	K-NN	4800 kannada characters	90%	Pre-segmented data, Susceptible to noise and outliers
[18]	Eccentricity and metric Parameter	K-NN	520 images	85%	Susceptible to noise, More prediction time
[20]	Otsu's thresholding Laplacian Filter	K-NN	4000 training, 78000 test set	96.8	Susceptible to noise, Requires Feature Scaling
		Ada-Boost Classifier		98.6	More training time, better prediction speed, less susceptible to noise
[21]	2D- Complex Wavelet Transform	ANN	MNIST Numerals	Around 99%	Implemented only on numerals
[12]	Discreet Wavelet Transform	ANN	4350 samples of Kannada characters	90%	Pre-segmented data
[4]	Horizontal and Vertical Wavelet Coefficients	CNN	4428 characters	96.58%	Database pre-segmented into characters
[5]	CNN	CNN-biLSTM	1539 documents	8.5 Character Error Rate per line	Better separation of similar character based on neighbouring characters
[22]	CNN – AlexNet	ECOC	NIST	97.7%	Wary of vanishing gradients, Better results compared to basic CNN
	CNN-ZfNet			97.65%	

VI. CONCLUSIONS

This paper has discussed various approaches for implementation of a handwritten character recognition. A Table 1 has been provided, giving a brief overview of some of the promising approaches towards character recognition. In section II, various research papers relevant to the topic have been explored and briefly discussed. Each approach could be roughly divided into three parts, i.e., pre-processing, feature extraction and character recognition and each of them have been discussed. Since, scripts of different language have considerable difference and some properties can be more significant in one language and insignificant in another. Generally, adopting a hybrid of various feature extraction methods could show good results. Using neural networks like CNN for feature extraction, have even shown better results than other feature extraction methods as seen in Table I.

Many approaches like neural networks, K-NN and SVMs have been discussed for recognizing characters. It is observed the neural networks are frequently used as classifier and even for feature extraction. This can be

attributed to the fact that deep learning approaches like neural networks have shown promising results. Especially, convolutional Neural Networks, have been widely used and have shown great and consistent results. Neural networks making use of Bi-LSTM networks have shown better results, while being computationally cheaper and efficient. They could be used to implement full page character recognition.

REFERENCES

[1] Ptucha, R., Petroski Such, F., Pillai, S., Brockler, F., Singh, V., & Hutkowski, P. (2019). Intelligent character recognition using fully convolutional neural networks. Pattern Recognition, 88, 604–613. https://doi.org/10.1016/j.patcog.2018.12.017

[2] Plamondon, R., & Srihari, S. (2000). Online and off-line handwriting recognition: a comprehensive survey. IEEE Transactions on Pattern Analysis and Machine Intelligence, 22(1), 63–84. https://doi.org/10.1109/34.824821

[3] Yadav, M., & Purwar, R. K. (2018). Integrating Wavelet Coefficients and CNN for Recognizing Handwritten Characters. 2018 2nd IEEE International Conference on Power Electronics, Intelligent Control and Energy Systems (ICPEICES), 1160–1164. https://doi.org/10.1109/icpeices.2018.8897291

[4] Garg, N. K., Kaur, L., & Jndal, M. (2015). Recognition of Offline Handwritten Hindi text using middle zone of the words. 2015 IEEE/ACIS 14th International Conference on Computer and Information Science (ICIS), 325–328. https://doi.org/10.1109/icis.2015.7166614

[5] Chung, J., & Delteil, T. (2019). A Computationally Efficient Pipeline Approach to Full Page Offline Handwritten Text Recognition. 2019 International Conference on Document Analysis and Recognition Workshops (ICDARW), 35–40. https://doi.org/10.1109/icdarw.2019.40078

[6] Singh, N. (2018). An Efficient Approach for Handwritten Devanagari Character Recognition based on Artificial Neural Network. 2018 5th International Conference on Signal Processing and Integrated Networks (SPIN), 894–897. https://doi.org/10.1109/spin.2018.8474282

[7] Sridharamurthy S K, & Reddy, H. S. (2015). PCA based feature vector for handwritten Kannada characters recognition. 2015 International Conference on Emerging Research in Electronics, Computer Science and Technology (ICERECT), 423–428. https://doi.org/10.1109/erect.2015.7499053

[8] Rani, M., & Meena, Y. K. (2011). An Efficient Feature Extraction Method for Handwritten Character Recognition. Swarm, Evolutionary, and Memetic Computing, 302–309. https://doi.org/10.1007/978-3-642-27242-4_35

[9] Wakabayashi, T., Pal, U., Kimura, F., & Miyake, Y. (2009). F-ratio Based Weighted Feature Extraction for Similar Shape Character Recognition. 2009 10th International Conference on Document Analysis and Recognition, 196–200. https://doi.org/10.1109/icdar.2009.197

[10] Ahmed, N., Natarajan, T., & Rao, K. (1974). Discrete Cosine Transform. IEEE Transactions on Computers, C–23(1), 90–93. https://doi.org/10.1109/t-c.1974.223784

[11] El qacimy, B., Ait kerroum, M., & Hammouch, A. (2014). Handwritten digit recognition based on DCT features and SVM classifier. 2014 Second World Conference on Complex Systems (WCCS), 13–16. https://doi.org/10.1109/icocs.2014.7060935

[12] Pasha, S., & Padma, M. (2015). Handwritten Kannada character recognition using wavelet transform and structural features. 2015 International Conference on Emerging Research in Electronics, Computer Science and Technology (ICERECT), 346–351. https://doi.org/10.1109/erect.2015.7499039

[13] Candes, E. J., Demanet, L., Donoho, D. L., & Yinh, L. (2005). Fast Discrete Transforms. Cal Tech. https://doi.org/10.1137/05064182X

[14] Padma, M. C., & Pasha, S. (2015). Feature extraction of handwritten Kannada characters using curvelets and principal component analysis. 2015 IEEE International Conference on Image Processing (ICIP), 1080–1084. https://doi.org/10.1109/icip.2015.7350966

[15] Nasien, D., Haron, H., & Yuhaniz, S. S. (2010). Support Vector Machine (SVM) for English Handwritten Character Recognition. 2010 Second International Conference on Computer Engineering and Applications, 249–252. https://doi.org/10.1109/iccea.2010.56

[16] Singh, D., Khan, M. A., Bansal, A., & Bansal, N. (2015). An application of SVM in character recognition with chain code. 2015 Communication, Control and Intelligent Systems (CCIS), 167–171. https://doi.org/10.1109/ccintels.2015.7437901

[17] Hari Rachmawanto, E., Rambu Anarqi, G., Moses Setiadi, D. R. I., & Atika Sari, C. (2018). Handwriting Recognition Using Eccentricity and Metric Feature Extraction Based on K-Nearest Neighbors. 2018 International Seminar on Application for Technology of Information and Communication, 411–416. https://doi.org/10.1109/isemantic.2018.8549804

[18] Freund, Y., & Schapire, R. E. (1997). A Decision-Theoretic Generalization of On-Line Learning and an Application to Boosting. Journal of Computer and System Sciences, 55(1), 119–139. https://doi.org/10.1006/jcss.1997.1504

[19] Pandya, V. J. (2016). Comparing Handwritten Character Recognition by AdaBoostClassifier and KNeighborsClassifier. 2016 8th International Conference on Computational Intelligence and Communication Networks (CICN), 271–274. https://doi.org/10.1109/cicn.2016.59

[20] Zhang, P., Bui, T., & Suen, C. (2004). Extraction of Hybrid Complex Wavelet Features for the Verification of Handwritten Numerals. Ninth International Workshop on Frontiers in Handwriting Recognition, 347–352. https://doi.org/10.1109/iwfhr.2004.41
[21] Bora, M. B., Daimary, D., Amitab, K., & Kandar, D. (2020). Handwritten Character Recognition from Images using CNN-ECOC. Procedia Computer Science, 167, 2403–2409. https://doi.org/10.1016/j.procs.2020.03.293
[22] Bhagyasree, P. V., James, A., & Saravanan, C. (2019). A Proposed Framework for Recognition of Handwritten Cursive English Characters using DAG-CNN. 2019 1st International Conference on Innovations in Information and Communication Technology (ICIICT), 1–4. https://doi.org/10.1109/iciict1.2019.8741412.