Handwritten Character Recognition of Assamese Handwritten Recognition Using Convolution Neural Network

A DISSERTATION

Submitted in the Partial Fulfilment of the Requirement for the Award of the Degree of

MASTER OF TECHNOLOGY

In

VLSI & Embedded Systems

Submitted By

ZAHIRUL ISLAM (2K20/VLS/25)

Under the guidance of

Mr. PIYUSH TEWARI

Assistant Professor, ECE Dept.

DEPARTMENT OF ELECTRONICS & COMMUNICATION ENGINEERING

DELHI TECHNOLOGICAL UNIVERSITY, SHAHBAD DAULATPUR, DELHI, 110042

DELHI TECHNOLOGICAL UNIVERSITY

SHAHBAD DAULATPUR, DELHI-110042

DEPARTMENT OF ELECTRONICS & COMMUNICATION ENGINEERING

CERTIFICATE

I hereby declare that the Project Report titled **"Assamese Handwritten Recognition Using Convolution Neural Network",** which is submitted by ZAHIRUL ISLAM, 2K20/VLS/25 of Electronics and Communication Department, Delhi Technological University, Delhi in partial fulfillment of the requirement for the award of the degree of Master of Technology, is a record of the project work carried out by the students under my supervision. To the best of my knowledge this work has not been submitted in part or full for any Degree or Diploma to this University or elsewhere.

DATE:

Under the Esteemed Guidance of

Mr. PIYUSH TEWARI

ASSISTANT PROFESSOR ECE DEPT.

CANDIDATE’S DECLARATION

I ZAHIRUL ISLAM, 2K20/VLS/25 students of M. Tech (VLSI &EMBEDDEDSYSTEM), hereby declare that the project Dissertation titled **“Assamese Handwritten Recognition Using Convolution Neural Network ’’** which is submitted by me to the Department of Electronics and Communication Engineering, Delhi Technological University, Delhi in partial fulfillment of the requirement for the award of the degree of Master of Technology, is original and not copied from any source without proper citation. This work has not previously formed the basis for the award of any Degree, Diploma Associateship, Fellowship or other similar title or recognition.

.

Place : Delhi

Date:

ZAHIRUL ISLAM (2K20/VLS/25

ACKNOWLEDGEMENT

I thank the almighty for giving us the courage and perseverance to complete the minor project. This project itself is acknowledgment for all those people who have given us their heartfelt cooperation in making this project a grand success.

I would like to thank Head of the E.C.E. Department, **Prof. N S RAGHAVA,** for providing constant encouragement. Sufficient infrastructure and good environment in the college to complete our course.

I am greatly indebted to project guide **Mr. PIYUSH TEWARI,** Assistant Professor, Electronics & Communications Engineering, for providing valuable guidance at every stage of this project work. I am profoundly grateful towards the unmatched services rendered by him.

Our special thanks to all the faculty of Electronics & Communications Engineering and peers for their valuable advises at every stage of this work.

Last but not the least, I would like to express my deep sense of gratitude and earnest thanks to my dear parents for their moral support and heartfelt cooperation in doing the project.

Zahirul Islam

Abstract

The involvement of technology is reshaping our perception of the world. The inherent desire to understand human consciousness and intelligence has led to the widespread development of the fields like Artificial Intelligence and Machine learning. [1] Artificial intelligence has paved its way into most disciplines and blended in as an essential tool to boost efficiency and non-conventional enhancements. Linguistics is one such field; with the involvement of AI, communication and text extraction have become vividly easier. The presented work involves the development of one such application: Handwritten Text Recognition for the Assamese language. The presented work analyses text extraction from images and understanding by classifying it into proper categories for machines to understand it using the Assamese language, which is spoken in the Indian state of Assam. The framework of the work can easily be utilized for other languages just by scanning or capturing the text of the mentioned language.

In this Project, the use of convolution neural networks(CNNs) is analyzed and proposed as the feature extractor for the handwritten Assamese characters. The classification for successful recognition of the scripts is achieved using the final layer of the CNN as a softmax activation layer. The dataset is obtained from the UCI repository for the training and testing of the proposed model. The results achieved by the testing of the model are quite satisfactory, with an accuracy of 99.87%.

TABLE OF CONTENTS

List of Figures

List of Tables

CHAPTER-1

Introduction

Optical Character Recognition (OCR) [2] identifies text within pictures with different recognition methods. OCR technology is used to turn almost any image with written text into machine-comprehensible text-formatted data. Handwriting OCR is able to reproduce the way people read handwriting, thanks to a mix of highly trained machine learning models and computer vision engines. The technology is helpful in numerous tasks, including digitizing paper-based records and detection of signature forgery.

1.1 Handwriting Recognition

The most important use of OCR systems is handwriting recognition [3]. Computer vision, machine learning, pattern recognition, image processing, and deep learning are all used in handwriting recognition. There are various forms of handwriting recognition like digits, characters, words, or signatures.

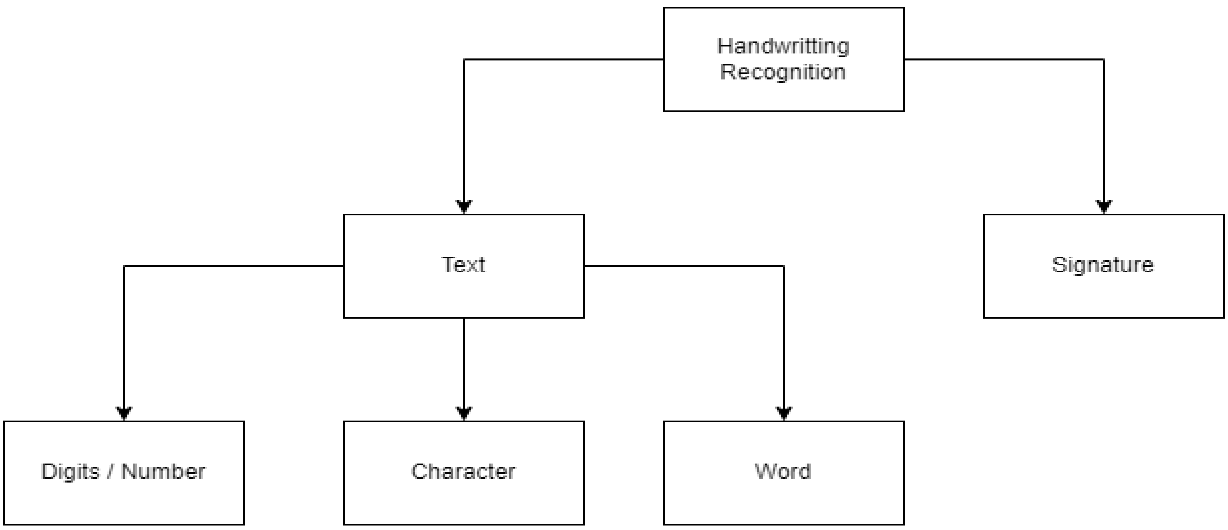

Fig. 1.1 Types of Handwriting recognition [47]

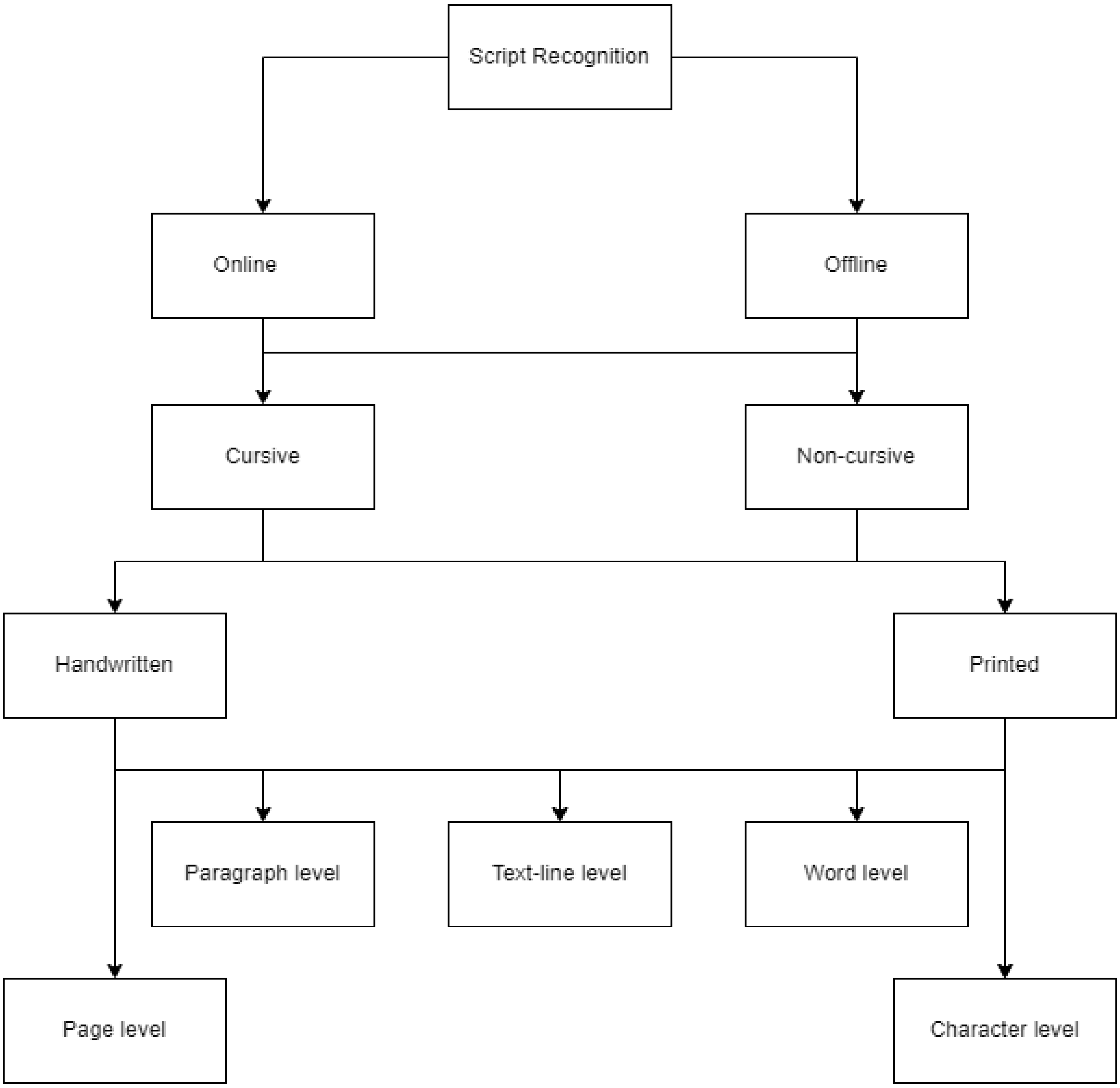

Fig. 1.2 Framework of the script recognition [48]

1.2 Classification

There exist mainly two forms of handwriting recognition methods mainly; online and offline.

The offline handwriting recognition [4] is implemented when the text is written on a hard copy which is scanned through devices like scanners and cameras. The text once captured goes through OCR (Optical Character Reader) [2] system which makes it digitally editable and suitable for recognition tasks.

The online system of handwriting recognition involves devices where the input can be taken in real-time using a touch screen or stylus. The main focus of the paper is in the offline method.
The offline handwriting recognition system extracts and reads any old documents like postcards, letters, signatures, banking documents, and manuscripts.

The offline recognition [5] systems are also split into two further categories one which involves segmentation and the other without segmentation. The techniques involving segmentation use methods to analyze the text by dividing it into its constituents like sentences, words, and individual numbers or characters. The predictive model then combines the results and tries to identify the complete words. The techniques without segmentation involve methods to recognize words as an entity and the method is often called HWR (Handwritten Word Recognition).

1.3 Applications

There are various popular domains where efficient handwriting systems are:

a. Finance:

Finance is an important applications of handwriting recognition systems. The finance sectors include banks where the rate of handwriting forgery cases can be monitored and reduced. The forms filled on paper by the elderly people with low interaction with electronic devices can be digitized using such recognition systems.

b. Health

In the hospitals the prescription of the doctor can be digitized in order to maintain a digital record of the patient's medical and medication history for future reference. The document can be shared directly with the chemist for proper circulation of the drugs in society.

c. Documents / Manuscript

Digitizing the documents from various public or private offices in order to store them on the computer or cloud for further processing or maintaining the database. The manuscripts of ancient scriptures can also be digitized and reproduced for analysis of content without any harm to the actual manuscript.

d. Education

Creating a digital copy of handwritten notes and digitizing the contents on the blackboard in the schools can reduce the time copying content from the blackboard and focus more on learning and understanding. The digital content once created can be distributed and reproduced easily.

1.4 Challenges

There are various challenges while developing methods to recognise handwritten characters from images. The challenges begin from the first step itself, data acquisition to the final step, classification. The challenges involve dealing with the noise in the captured image or low-quality and old images. There are also different variations to deal with when considering handwriting, the difficulty just spikes with the introduction of cursive writing styles into consideration. The modifiers should be in writing in a particular order and one of the most troublesome while creating a predictive model is the lack of good and labeled datasets.

1.5 Organization of the report

The proposed report consists of the following sections. Chapter 2 gives a brief idea about Assamese handwriting recognition, the chapter also includes script, challenges, datasets, and a general framework for the handwriting recognition tasks. Chapter 3 analyses the literature review of the techniques of ML and DL implemented by numerous researchers regarding the recognition of the Assamese handwritten characters. The chapter 4 analyses the incorporates the proposed recognition system. The entire experimental details are explained in the chapter 5 and the results are described in the chapter 6. Finally, chapter 7 drafts a conclusion from the experiments. Chapter 8, ends the report with the future outcomes for the betterment of the presented work in the concerned domain. The last section mentions all the references used.

CHAPTER-2

Assamese Handwriting Recognition

2.1 Assamese Language

The language is mainly spoken in the northeast Indian state of Assam [6]. It's the official language of Assam. The language is spoken by over 23 million speakers. The language is derived from the Assamese script. The Assamese script is derived from the Brahma script and is evolved for years.

The Assamese language has Forty consonants, Eleven vowels, ten modifiers, and nearly 300 compound characters. There exits the use of headline (matra) in characters including vowels and consonants.

In Assamese, upper and lower case letters are not utilised as they are in English. Every person has their own writing style. This style varies depending on the individual and their mental condition. The writing style of a person varies as they get older.

Fig. 2.1 The Vowels

A typical Assamese word can be categorized into three different zones.

- Above the headline, Upper Zone It is distinguished by the presence of modifier extensions.
- The character's primary body is located in the Middle Zone.
- Lower Zone Area, which has some modifications

Fig. 2.2 The Consonants

2.2 Challenges

There are various challenges to address when working with the Assamese script [7]. The primary challenge to address is the lack of a dataset to work with. The dataset mentioned in the next section is the only dataset available. There is a serious lack of well-formulated, labeled Assamese characters datasets. Other challenges come when the images captured have noise, noise removal can be one of the major challenges as well. The writing styles also impact the slants and strokes of the characters in the Assamese script. There are a lot of variations for an individual character but not enough dataset for models to train and understand.

2.3 Dataset

The UCI repository [9] provided the data for the Assamese characters. There are around 8326 pictures that match to different letters, combining letters (zuktakkhor), and numerals. The initial data consisted of lines and strokes used to write each alphabet, rather than pictures.

Assamese is the Indo-European language family's most eastern member. Letters may be concatenated to make compound alphabets, much like in Hindi. There are around 183 classes in the dataset, which encompasses virtually all possible combinations. This results in less than 50 photos for each class, which is less than ideal. This is the only complete collection of handwritten Assamese characters that exist.

2.4 Framework

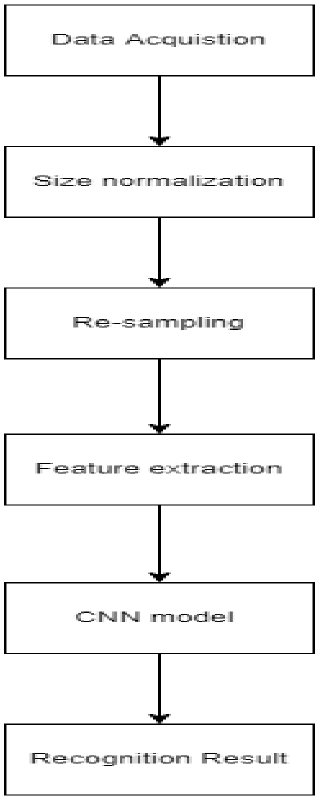

Fig. 2.3 Offline HRS Framework

The stages of digital handwriting recognition systems:

1. **Data Acquisition:**
   The handwritten scripts which are intended to be recognized by the system should be digitized for the model to train. The process is accomplished using devices like scanners or cameras (modern smartphones).

2. **Pre-processing:**
   The step includes cleaning the images with noise using various computer vision and image processing techniques involving applying filters like median, Gaussian, etc. The normalization of the dataset is crucial for the model to understand the content in the images. The data is read by the model in the form of the pixel values which are normalized.

3. **Segmentation:**
   Before proceeding with training and testing of the neural network, the scanned document is segmented into lines, words, and characters.

4. **Recognition:**
   The predictive model is used for learning the features of the characters of the scripts and once the learning of those features is achieved then the classification of the characters into their respective labels is done.

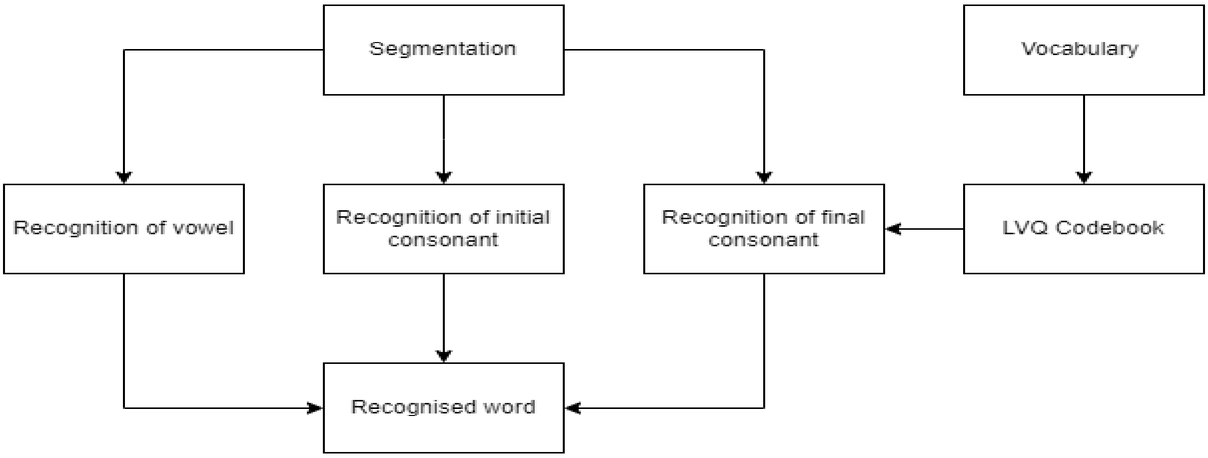

Fig. 2.4 Segmentation of scripts for recognition

CHAPTER-3

Literature Review

3.1 Review of Machine Learning Algorithms based handwriting recognition system:

This paper [12] focuses on the creation of an Assamese online character recognition system utilising HMM and SVM, as well as a comparison of the two models' recognition capabilities. Training 181 different Assamese Stokes produced recognition models using HMM Toolkit and SVM Toolkit. The testing for stroke and Akshara levels are done independently.

In this paper [14] a Hidden Markov Model is used to train 181 various stroke patterns, resulting in a model that may be utilised for stroke level assessment. Using a dynamic linked library, Akshara level testing is conducted by combining a GUI (supplied by CDAC-Pune) with the Binaries of HTK toolkit classifier, HMM train model, and language (dll). They scored 94.14 percent on the stroke test and 84.2 percent on the akshara test.

In this paper [44] the creation of a recognition model using the Hidden Markov Model and Support Vector Machines is described in this paper (SVM). Both modeling strategies employ coordinates and their first and second derivatives at each place is analysed.
HMM and SVM are used to create the two systems separately. After then, the findings from both systems are integrated using two separate methods.

It is beneficial in a wide range of real-world applications, isolated character recognition has been the topic of extensive research in recent decades. It also has a solution for automatically processing massive amounts of data.The paper [45] presents a research on the Gurmukhi script and analyses it with various machine learning algorithms.

3.2 Review of State-of-the-Art Deep Learning-based recognition systems:

In [9] this paper explains how to divide characters for an ANN-based character recognition system that recognises optically scanned handwritten Assamese characters. Character segmentation is accomplished by projecting the handwritten text document horizontally and vertically. The method extracts the geometric features of the characters, which are made up of fundamental line types that are utilised to build the character skeleton, for feature extraction. The recognition system is trained using ANN utilising the feature vector of the training set generated by this system.

In this paper [10], A CNN-based Bangla handwritten character recognition is researched in this work. The suggested technique uses CNN to categorise individual characters after normalising the written character pictures. Unlike previous comparable efforts, it does not use a feature extraction approach. In this investigation, 20000 handwritten characters of various forms and variants were employed.

The authors of this study [11] explore several unique online handwriting and machine-printed text recognition algorithms in Assamese. Unlike English and other languages, the Assamese language is mostly cursive. The basic objective is to learn how to write in cursive.

This paper [13] investigates the use of ANN to help in the segmentation of handwritten characters in Assamese, a major language in India's North East. The study compares the performance of an ANN-based dynamic segmentation algorithm to that of a projection-based static segmentation technique.

In this paper [16] the terms of adversarial resilience, MNIST appears to be far from being solved.They introduce an unique robust classification model that uses learnt class-conditional data distributions to perform analysis via synthesis.

In this paper [17] the first comprehensive and up-to-current evaluation of this dataset; there are some online rankings, but they are out of date, and most published articles only examine closely related works, leaving out the majority of the literature. This study

distinguishes between works that use data augmentation and those that use the original dataset straight out of the box.
[18] Aims to learn and interpret the various forms of handwritten digits, this study uses a convolutional neural network model (DigiNet). Our model employs a convolutional neural network with six different Convolution and Pooling layers to achieve state-of-the-art performance on Assamese handwritten digits, with a test accuracy of 93.02 percent.

In this [20] research, authors suggested utilising a multilayer perceptron with one hidden layer to detect handwritten Tamil letters. Fourier Descriptors are the feature retrieved from the handwritten character.

The paper [40] involves creating a combined character and numeric recognition system employing a hybrid feature set that was previously used to recognise Assamese characters. The usage of a modified hybrid feature set created for handwritten number detection using a neural network is also described in the paper. The set seeks to increase identification performance, resilience, and shape and size invariance in the presence of noise.

In this paper [41] the aim of the authors is to recognise the characters using a feed-forward back propagation neural network with two hidden layers. With the limitation of a less cursive writing style, two types of data for Assamese scripts were obtained from various people. Isolated characters fall into one group, while sentences fall into another.

In this paper [43] aim is to apply the same enhancement strategies to CNN units that we've had success with Stacked Denoising Auto-Encoder classifiers.
When categorising MNIST digits, this results in a new performance record. The auto-encoders are used to denoise the data.

The goal of this [46] study is to use a feed forward neural network to create a recognizer for Assamese digits. The recognizer extracts the feature by cropping the individual digits of the picture using the bounding box approach.

Table 3.1 Summary of Literature Review and related works

Author(s) Name	Reference	Year	Method used	Results
Sagarika Borah	[8]	2015	Artificial Neural Network	Tested with 10 samples with 78% accuracy
Md. Mahbubar Rahman, et al	[10]	2015	BHCR-CNN	Accuracy 85.96%
Keshab Nath, Subhash Basishtha	[11]	2015	HMM and SVM	-
Deepjoy Das, et al.	[12]	2014	HMM and SVM	The average recognition accuracy in HMM was 94% and in SVM, 96%
Kaustubh Bhattacharyya et al	[13]	2009	ANN	MSE for 0.008x10^-3 and accuracy 95.5%
SRM Prasanna, et al	[14]	2014	HMM	Stroke level performance 94.14% and akshara level performance 84.2%
Debal Pedamonti [15]	[15]	2018	-	-
Lukas Schott, et al	[16]	2018	CNN	Accuracy 99%
Alejandro Baldominos, et al	[17]	2019	-	-
Prarthana Dutta, Naresh Babu Muppalaneni	[18]	2021	CNN	Accuracy: 93.02%

Nilakhi Saikia, S.R. Nirmala [19]	[19]	2018	ANN, HMM	Best recognition achieved: 97.28%
J.Sutha, N. Ramraj	[20]	2007	Multilayer perceptron	Accuracy: 97%
Kandarpa Kumar Sarma	[40]	2007	Multilayer perceptron	Accuracy: 96.2%
Rubul Kumar Bania, RuhUllah khan	[41]	2018	Gradient descent with momentum training and adaptive learning	Accuracy: 90.34%
Ernst Kussul, et al [42]	[42]	2004	LImited Receptive Area (LIRA)	Accuracy: 99.41%
Ricardo F. Alvear-Sandovala, Jos´e L. Sancho-Gomezb, An´ıbal R. Figueiras-Vidala	[43]	2018	CNNs	-
Bandita Sarma, et al	[44]	2013	HMM, SVM	Accuracy: 96.5%, 96.8%, 98.3%
Dharamveer Sharma, Puneet Jhajj	[45]	2010	KNN, SVM	Accuracy: SVM: 73.02%
Kalyanbrat Medhi, Sanjib Kr. Kalita	[46]	2015	ANN	Accuracy: 84.3%

CHAPTER-4

PROPOSED SYSTEM FOR ASSAMESE HANDWRITTEN CHARACTER RECOGNITION

4.1 Objective

To obtain information from clients, most businesses employ paperwork. Typically, these documents are handwritten. Forms, checks, and other similar paperwork fall within this category. Documents are converted and stored in digital formats for quicker retrieval and information collection. Manually entering the same data into a computer is a common approach for dealing with that information. Handling such documents manually would be exhausting and time consuming. As a result, a specific Handwritten Character Recognition Software is required, which will recognise texts from images of documents automatically. Handwritten Character Recognition (HCR) Software has simplified the process of extracting data from handwritten papers and storing it in electronic formats. Handwritten documents are frequently utilised in banking, health care, and other areas. HCR systems are also useful in areas where handwriting data entry is necessary, such as the creation of electronic libraries and multimedia databases.

The aim of the proposed method is to automate the handwritten character recognition procedure and to avoid the struggle and hassle of doing this manually. While doing this, the classification accuracy of the recognition must increase, and the recognition time lapse must decrease.

The motivation of the proposed method is to:

- Minimize the image pre-processing requirement
- Elimination of the requirement of extraction of engineered feature selection and extraction.
- Improve recognition accuracy.
- Reduce prediction time

Proposed Recognition system

The proposed recognition system is described in this section. Different steps and related theory has been described in this section.

The Dataset

This UCI repository provided the information for the Assamese characters[21]. There are around 8326 images that match to different letters, conjoined letters (zuktakkhor), and numbers. The initial data consisted of lines and strokes used to write each alphabet, rather than images. Because this will not be useful for character recognition, I have converted the data to photos and divided it into train and test sets, making it easier to utilise Keras for classification. To assess the accuracy, I'll use a Convolutional Neural Network as a first trial. The images consists of Assamese characters each of 32*32 pixels in RGB color format.

Pre-processing

The images are converted to binary and normalization is of the data is done by dividing them by 255. Data augmentation is also implemented on the image. The images are zoom in by 30% and sheared by 30%. The image are resized to 50*50 pixels before implementing the classification algorithm.

Feature Extraction using CNN

Instead of implementing the feature extractor manually, CNN implements it in the training phase. The CNN feature extractor is composed up of different categories of neural networks that regulate the model weights during the training of the CNN. When CNN's neural network feature extraction goes deeper with an increased number of layers, it improves picture identification at the expense of the learning technique complexity that had previously rendered CNN inefficient and disregarded. CNN is a category of neural network that extracts visual features and classifies them using other variant of neural network. The feature extraction network of the CNN leverages the input image. The neural network uses the extracted feature information for classification. The output is then produced by the neural network classification, which works on the picture features. A collection of a number of convolution layers and pooling layers are included in the neural network for feature extraction.

Classification:

In automatic image classification and pattern recognition systems, convolutional neural networks (CNN) are commonly employed. Typically, the upper layers of the CNN is utilized for classification; however, those features may not include enough useful information to properly predict an image. Lower-layer characteristics have a stronger discriminative power than upper-layer features in some cases and are thus used in those situations. Convolutional neural networks changed image classification by learning basic forms in the early layers and growing to learn picture attributes in the deeper layers, resulting in more accurate image classification. A CNN is made up of several convolutional layers and a number of subsampling layers. They are followed by fully linked output layers if desired. The model is trained using the back propagation algorithm.

Performance Evaluation

The classification performance of the proposed CNN model is evaluated using classification accuracy and categorical cross-entropy loss. These have been described in the following sections.

4.2 The CNN Model

Convolutional Neural Networks (CNNs) are homogenous to classic artificial neural networks (ANNs) where they consists of neurons that learn to optimise themselves. Each neuron in the network receives an input and perform an operation (such as a scalar-product followed by any non-linear function) - the basic building blocks of almost all ANNs. The entire network expresses a single perceptual scoring function from the raw input image vectors to the probability score of the final output classes. The last layer consists of loss functions specific to the number classes in the problem.

The primary crucial dissimilarity between CNNs and standard ANNs is that CNNs are mainly employed for recognition of pattern in images . This network allows encoding of image-specific properties and features into the CNN architecture, making the neural network better suited to tasks focussed on images while also lowering the number of parameters needed to set up the model.

Convolution Layers

The convolutional layer is crucial to how CNNs operate, as its name indicates. The layers settings are focused on the learnable kernels. These kernels have a limited spatial dimensions, yet they extend throughout the wholeinput depth. When data vectors progress through a convolutional layer of the CNN, the layer convolves each filter across the input's spatial dimensions to generate a 2D activation map[22]. The scalar-product is calculated for each value in that specific kernel as we progress through the input vectors. The network will learn specific kernels that procduced output when they see a specific feature at a specific spatial position in the input as a result of this. These are also known as activations maps.

The convolutional layers of CNNs greatly reduce the model's compuational complexity by optimising its output. The depth, stride, and establishing zero-padding are three hyperparameters that are optimised[22]. The number of neurons in the layers can be tuned manually to the concerned region of the input image to fix the depth of the output volume fabricated by the convolutional layers. This may be seen in many types of ANNs, where all of the neurons in the hidden layer are previously directly coupled to each other. Minimizing this hyperparameter reduces the total number of neurons in the network, but it also reduces the model's ability to recognize patterns.

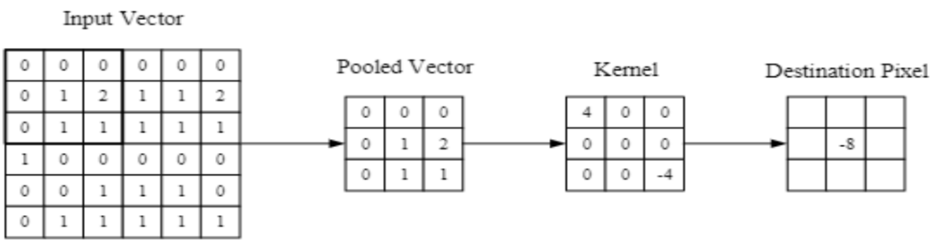

Fig. 4.1 The feature extraction process of a convolutional neural network using kernels.

To determine where the receptive field should be placed, we can also determine the stride where the depth is defined specific to the input's spatial dimensions. If we assign the stride to 1, for example, we will have a substantially overlapping receptive-field with extraordinarily big activations. Alternatively, increasing the stride will limit the amount of overlapping and result an output with lower spatial dimensions. Zero-padding is a basic process of padding the input's boundary, and it's an excellent way to manage the dimensionality of the output volumes.

Pooling Layers

The goal of pooling layer is to gradually diminish the representation's dimensionality., This is achieved by carefully lowering the the number of hyper-parameters and subsequently the complexity of the model[22]. The pooling layer bring down the dimensionality of each activation map in the input vector. These are usually max-pooling layers with szie of kernels 2*2 applied with a stride of value 2 throughout the spatial dimensions of the input feature maps in most CNNs. The size of the activation maps is reduced to 25% of its original size while the depth volume is maintained at its full size.

Due to the fact that different pooling layers are damaging, only two types of max-pooling functions are usually observed. The stride and filters of the pooling layers are usually both set to 2 2, which allows the layer to stretch over the input image’s spatial dimensions[22]. Overlappingpooling can also be used, with the stride value of 2 and the kernel size of 3. Due to the diminishing effct of pooling function, employing a kernel size greater than 3 will significantly reduce the performance of the models.

It's also vital to note that, in addition to max-pooling layers, CNN architectures can also include general pooling layers. Poolingneurons in normal pooling layers are capable of performing a variety of typical operations like as L1/L2-normalisation function and also average pooling function.

Dropout Layers

The Dropout layer, which helps prune the effect of overfitting of the model, at each step throughout training period, randomly sets input units to 0. Inputs that are not put to 0 are scaled up by 1/(1 - rate) to keep the total sum unchanged.

Fully Connected Layers

The fully-connected layer consists of neurons that are directly linked to neurons in the preceding and succeeding adjacent layers, but not to neurons in any of the non-adjacent leves[22]. This is similar to the manner neurons are organised in standard ANN.

TABLE 4.1. DETAILS OF THE LAYERS OF A CONVOLUTIONAL NEURAL NETWORK

Layer Type	Functions	Parameters	Input	Output
Convolution Layers	• Convolution of filters with images o generate feature maps • Filters are made of kernels having bias 1 per filter • Activate all values in feature maps	• No. of kernels • Size of kernel(height and width) • Activation function • Stride size • Padding value • Value and type of regularization	3D array, prior feature maps	3D array, 1-2D feature map for every filter
Pooling Layers	• Dimensionality reduction • Extract average or max pixel value from a region • Approach of sliding window	• Stride size • Size of window	Three dimensional array, prior feature maps	Three dimensional array, 1 two dimensional feature maps, Spatial dimensions diminished
Fully Connected Layers	• Sum of output data from every feature maps	• No. of nodes • Activation function: if aggragating info, use	Flattened cuboid, previous feature maps sets	Three dimensional cube, one two dimensional

	• Classification into classes	ReLU, else for final output layer, use softmax		map per filter

In neural networks, activation functions determine the weighted sum of input vector and associated biases, which is used to check whether a particular neuron can compute output or not. It uses gradient processing, most frequently gradient descent, to alter the input data before generating the output of the neural network that comprises the parameters of the input data. In certain literature, these activation functions are also known as transferring function.

Activation functions are utilised to govern the outputs of our neural networks across different domains and these be either linear activation function or non-linear activation function depending on the operation they represent.

These AFs are required to convert input signals that are linear and models them into non-linear output signals, which clear the way for the deeper neural networks, and assist in the efficient learning of high-order polynomials. Non-linear activation functions possess the distinct advantage of being a differentiable function; otherwise, they would not operate during backpropagation in networks that extends to deep layers.

ReLU Activation Function

Nair and Hinton proposed the rectified linear unit (ReLU) activation function, and it has been the most extensively employed AF for deep learning applications that provide cutting-edge outcomes [23]. The ReLU is a faster learning activation function[24] that has shown to be the most popular and successful[25]. In deep learning, it outperforms the previously used Sigmoid and Tanh AFs[26][27] in terms of generalization and performance. The ReLU represents a not exactly a linear function, but keeps the characteristics that generally defines a linear models and that make gradient-descent techniques possible[28] and it is also easy to optimise.

The ReLU AF applies a threshold value to every input, setting the values less than 0 to zero.

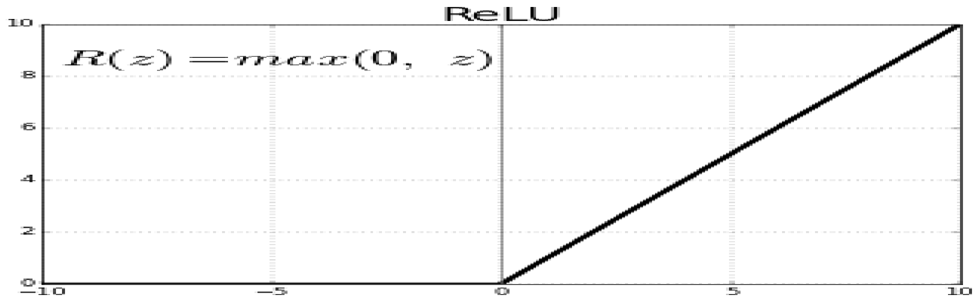

Fig. 4.2 The ReLU activation function

This AF corrects the values and eliminates the problem of vanishing gradient that was present in earlier forms of AFs. With typical instances seen in object classification, the ReLU has been applied in the hidden units of other deep neural networks.

The fundamental benefit of utilising rectified linear units in the calculations of neural output is that they ensure faster computation of the neurons since they don't perform divisions operations or exponential operations, resulting in a higher overall performance in the neural computations. This AF also compacts the value between 0 and maximum value, leading to sparsely distributed hidden units. However, the ReLU has the disadvantage of readily overfitting when evaluated and compared to the sigmoid function, despite the fact that the dropout approach has been used to lessen the overfitting in this AF and the rectified networks have enhanced deep neural network performance.

Softmax Activation Function

Another form of activation function utilised in neural computing is the Softmax function. It's used to produce a probability distribution out of a set of real numbers. The output of the this function is a range of values between 0 and 1, with the sum of the probabilities euallling to 1.

It has most common application in multi-class classification where the Softmax function returns probabilities for each predefined classes, where the target class has the highest likelihood. Almost all output layers of the deep learning architectures for various applications [29][30][31] use the Softmax function.

Categorical Cross-Entropy

The loss function cross-entropy is commonly used in machine learning algorithms. Cross-entropy is a metric evaluates the difference between two provided probability distributions and is based on entropy. Cross-entropy calculates the total entropy between the probability distributions.

This function is commonly mistaken with logistic loss, most commonly referred to as log loss. Despite the fact that the two measures are acquired from distinct sources, when employed as loss functions in classification models, they are the same amount and may be used interchangeably.

In almost every multi-class classification tasks, categorical crossentropy is the most commonly employed loss function.

Adam Optimization Function

Gradient Descent is useful in situations where the function is easily differentiable in relation to the network parameters. It is easier to minimise continuous functions than discrete functions, according to. After one epoch, the weight is updated, with one epoch representing running through the complete dataset. This approach delivers acceptable results, but it degrades and does not converge effectively as the training dataset grows larger. In the case of many local minima, it may not lead to a global minimum. By sampling random data samples and changing the hyper-parameters based on the cost function, stochastic gradient descent avoids this flaw. It also converges faster than traditional gradient descent and saves memory by avoiding the accumulation of intermediate weights. The suggested Adaptive Moment Estimation (ADAM) method simplifies the estimation of learning rates for each parameter by utilising the first and second moments of gradient. The same has been summarised and discussed. ADAM uses little memory and outperforms on huge datasets since it is computationally efficient.

CHAPTER-5

EXPERIMENTAL SETUP

The proposed methodology is carried out using python programming language on Anaconda Jupiter Notebook. More details about the experimental setup is discussed in this section.

5.1 Installation of Libraries

Pandas

Pandas is an open-source library thet was developed for easy handling of relational data or labelled data[32]. It provides with quite a many data-structures and related methods which help in manipulating numerical data as well as time series data. With this library is based on the previously developed NumPy. It is very fast and provides users with excellent performance and productivity.

Numpy

NumPy, or Numerical Python, is a python-based library that contains multi-dimensional array objects[33] and a collection of associated functions and methods for data manipulation. Functionality provided by it involves mathematical and logical computation on arrays.

The most significant object in NumPy is the ndarray type of N-dimensional array. It refers to a group of items of the same type. A zero-based index can be used to find items in the collection.

Scikit Learn

In Python, Scikit-learn (Sklearn) is the most popular and robust machine learning toolkit for python programming language[34]. It provides a set of efficient machine learning and statistical modelling capabilities, including as classification, regression, clustering, and dimensionality reduction. It mostly focuses on machine learning techniques, such as mathematical, statistical, and general-purpose algorithms, which provide as the foundation for numerous machine learning technologies.

Tensorflow

TensorFlow is one of the most important framework for machine learning and deep learning. The mentioned framework is developed and maintained by google. The framework reduces human effort and widens the use of the ML models which was priorly was only limited to core ML researchers. The ease of use and ready to use templates allows to create and tweak the models as per the requirements. Most of the functions like loss and optimisation are available in the form of methods. It simplifies the experience of beginning the journey in the field.

Keras

Keras is a user-friendly API built over the Tensorflow framework [36]. Keras follows best practices for reducing cognitive load, such as offering consistent and straightforward APIs, limiting the number of user activities necessary for typical use cases, and displaying clear and responsive error signals. It can scale to massive clusters of GPUs or a full TPU pod. It's not only possible but also simple. Keras is an important part of the TensorFlow 2 ecosystem, since it handles every stage of the deep learning workflow, from managing data to hyperparameter tuning and deployment strategies.

Matplotlib and Seaborn

One of the most commonly used library for easing data visualisation is Matplotlib[37]. It's a cross-platform toolkit for drawing 2D charts out of array data. It also works with Jupyter notebooks and web app servers, as well as Python and Anaconda shells.

Seaborn is based on matplotlib and integrates with pandas dataframes[38]. Its plotting attributes deal with dataframes and arrays spanning full datasets, internally executing the relevant semantic mapping and statistical aggregation to deliver quality graphs. Because of its declarative, dataset-oriented API, users can focus on the meaning of your charts rather than just the mechanics of producing them.

OpenCV

OpenCV (Open Source Computer Vision Library)[39] is a public computer vision and machine learning software library. It was developed to offer a standard infrastructure for computer vision applications and to make it much easier for commercial applications to incorporate machine vision.

Over 2500 optimised algorithms are included in the collection, which span the entire spectrum of classical and cutting-edge computer vision and deep learning methodologies. Businesses, academic researchers, and governmental organizations all use the library. It leans heavily toward real-time vision applications. This library is implemented in C++ and has a templated user framework that works really well STL containers.

5.2 Dataset Preparation

The dataset has been divided into two parts for training of the CNN classifier. The two parts are training and validation set. Each of these set contains 402 character samples. The Assamese language has 40 consonants, 11 vowels, ten modifiers, and nearly 300 compound characters. The images are converted to binary and normalization is performed by dividing them by 255. Data augmentation is also implemented on the image. The images are zoom in by 30% and sheared by 30%. The image are resized to 50*50 pixels before implementing the classification algorithm.

Algorithm:

1. The training and validation dataset is read from its respective directories.
2. The pixels of each image samples are normalized by diving the pixels by 255.
3. Data augmentation is applied wherein the images are each zoomed and sheared by 30%.
4. The input image samples are scaled to 50*50*3.
5. The output values are converted into categorical feature vectors.
6. The CNN is implemented on the training set and validated using validation set and trained for 100 epochs.
7. The performance of the training is evaluated using classification accuracy, confusion matrix and loss value.
8. The performance is also evaluated using unknown test data.

5.3 Creation of the CNN Model

The CNN model is created using sequential API of Keras backend. The model layers are added using add() function. There are total 920,823 parameters and all of them are trainable parameters.

The layers implemented in the model is described below:

input: A 2D convolution layer is added having 32 filters and a kernel size of 3*3 with ReLU activation function.

max_pooling2d: Maximum Pooling layer added for subsampling the size of the feature maps. Pool size of 2*2 is used.

conv2d: A convolution layer is added having 32 filters and a kernel size of 3*3 with ReLU activation function.

max_pooling2d_1: Maximum Pooling layer added for subsampling the size of the feature maps. Pool size of 2*2 is used.

dropout: This layer is added to minimize the chances of overfitting. 25% of the units have been randomly dropped from the connections.

conv2d_1: A convolution layer is added having 64 filters and a kernel size of 3*3 with ReLU activation function.

max_pooling2d_2: Maximum Pooling layer added for subsampling the size of the feature maps. Pool size of 2*2 is used.

Dropout_1: This layer is added to minimize the chances of overfitting. 25% of the units have been randomly dropped from the connections.

flatten: It is used to convert the feature maps from the previous layers (5*5*64) into a 1-D vector of size (1600).

dense: The flattened layer is fed to a fully connected layer with 500 units.

dropout_2: 25% of the units have been randomly dropped from the connections.

dense_1: Last Fully connected layer has units equal to the number of classes in the training set. Here, it is 183.

```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
input (Conv2D)               (None, 48, 48, 32)        896
_________________________________________________________________
max_pooling2d (MaxPooling2D) (None, 24, 24, 32)        0
_________________________________________________________________
conv2d (Conv2D)              (None, 22, 22, 32)        9248
_________________________________________________________________
max_pooling2d_1 (MaxPooling2 (None, 11, 11, 32)        0
_________________________________________________________________
dropout (Dropout)            (None, 11, 11, 32)        0
_________________________________________________________________
conv2d_1 (Conv2D)            (None, 9, 9, 64)          18496
_________________________________________________________________
max_pooling2d_2 (MaxPooling2 (None, 5, 5, 64)          0
_________________________________________________________________
dropout_1 (Dropout)          (None, 5, 5, 64)          0
_________________________________________________________________
flatten (Flatten)            (None, 1600)              0
_________________________________________________________________
dense (Dense)                (None, 500)               800500
_________________________________________________________________
dropout_2 (Dropout)          (None, 500)               0
_________________________________________________________________
dense_1 (Dense)              (None, 183)               91683
=================================================================
Total params: 920,823
Trainable params: 920,823
Non-trainable params: 0
_________________________________________________________________
```

Fig. 5.1 Layers, parameters and number of units of the model

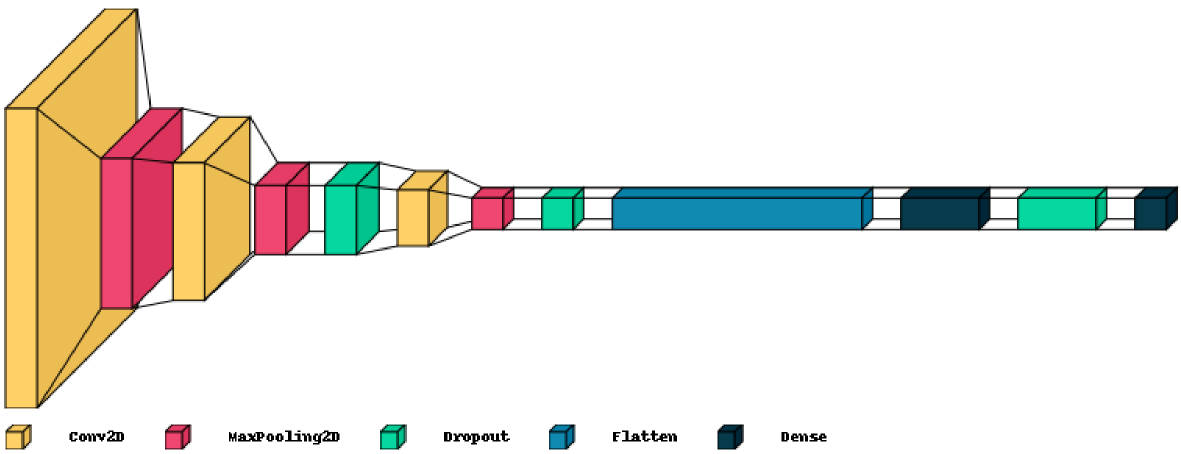

Fig. 5.2 The CNN model Architecture

CHAPTER-6

RESULT AND OBSERVATIONS

The CNN model was implemented on the mentioned Assamese dataset and it is evaluated using classification accuracy. The training accuracy of the model is 98.51% and the validation accuracy is 99.75%. This result have been obtained after training the model for 100 epochs. The training loss and accuracy of both training and validations set is presented in in the figures below.

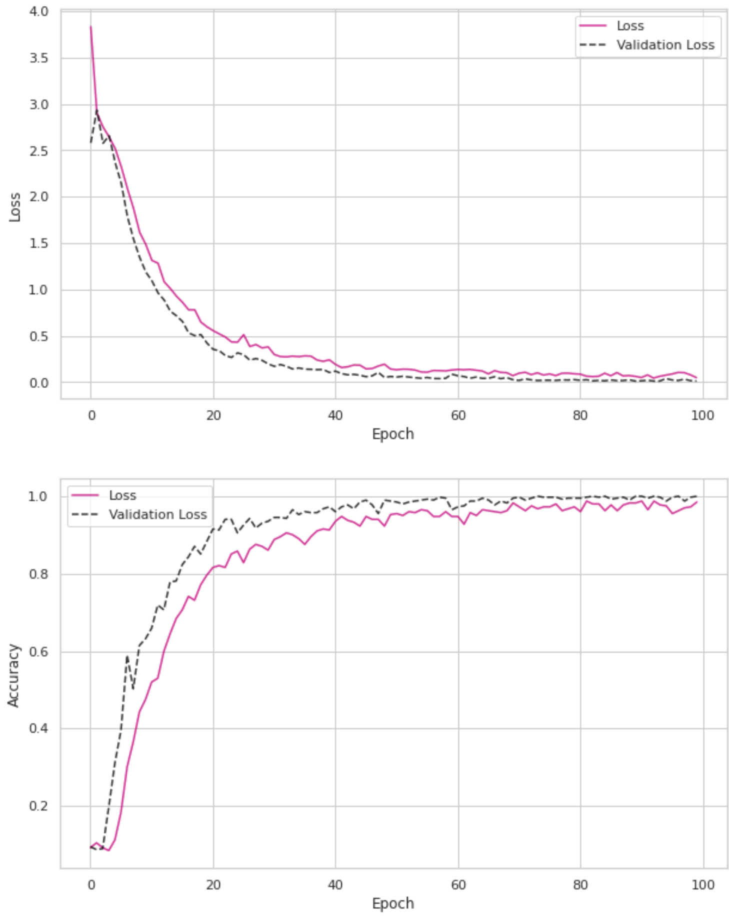

Fig. 5.3 (Top) Training and validation loss. (Bottom) Training and validation accuracy.

It can be observed that there is no overfitting in the model. The classification accuracy and loss of the training and validation dataset converges at the 100th epochs. The classification accuracy on the unknown test set is 99.75%. This shows that the model predicts with very good accuracy. The confusion matrix shows that there are very few misclassification of the predictions.

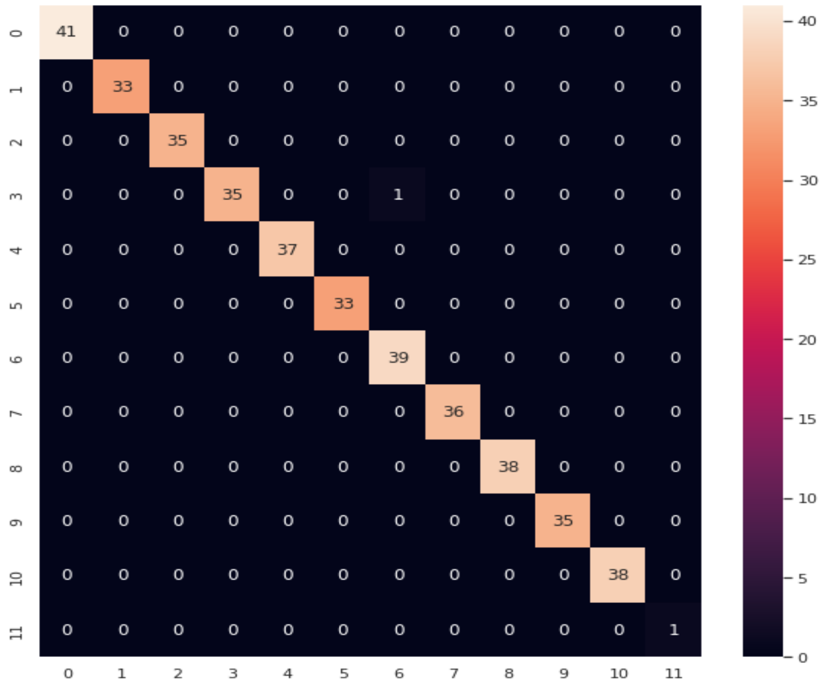

Fig. 5.4 The confusion matrix of the CNN model.

The performance of the CNN model was also evaluated on individual image samples from the test set. The model was able to predict most of the samples. The activation maps of the convolution layers of the CNN model is also visualized. These observation is presented in the figures below.

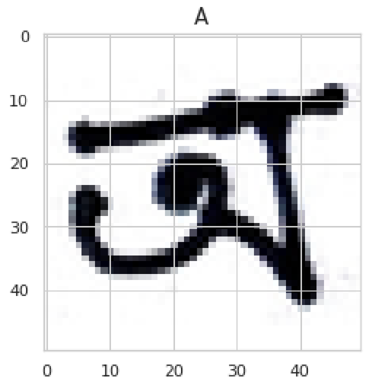

Fig. 5.5 Sample test image 1.

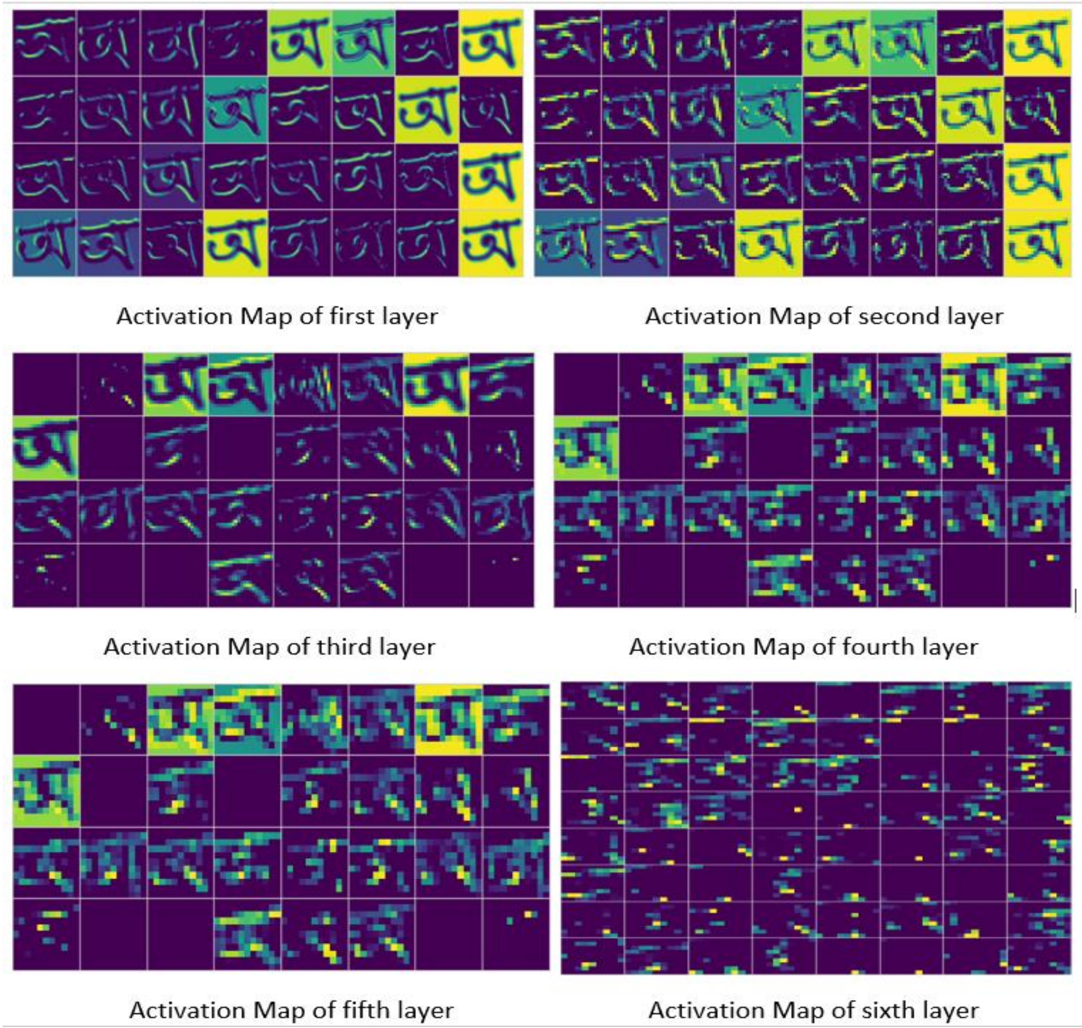

Fig. 5.6 Activation maps of different layers for sample test image 1.

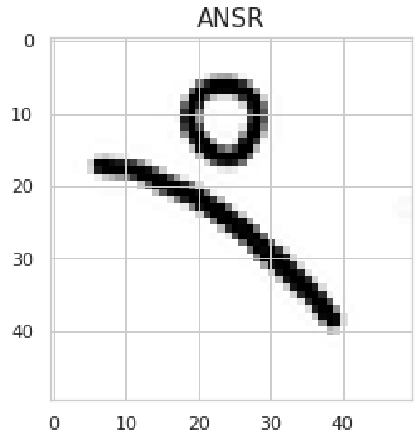

Fig. 5.7 Sample test image 2.

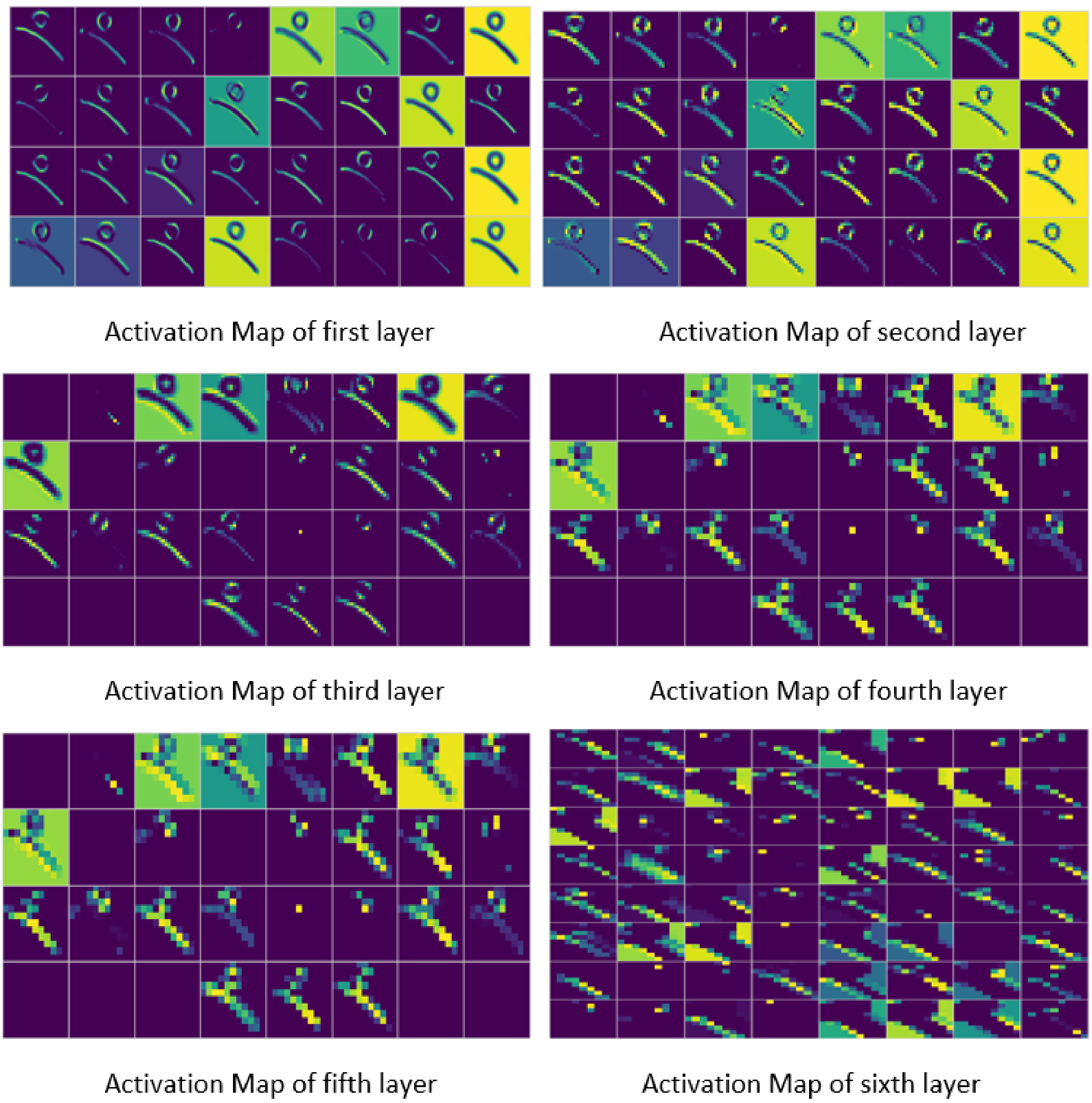

Fig. 5.8 Activation maps of different layers for sample test image 2.

The fig. 5.6 and 5.8 are the class activation maps for the sample images fig 5.5 and fig 5.7. The class activation map is used to find the discriminators in the image to classify them into a particular class. While training feature extraction decides features corresponding for the particular associated classes. The activation maps enable us to visualize the descriptors after each convolution layer. The breaking down of the image into its features to classify it.

CHAPTER-7

CONCLUSION

The implemented CNN model is able to accurately recognize Assamese handwritten characters. The classification accuracy of the model on the unknown test set is 99.75%. It is observed that the model does not overfit during the training as the loss and accuracy values converge. The CNN model is able to recognize key aspects in images, and thus when compared to typical pattern recognition approaches for feature extraction, this method saves time and money. It also eliminates the time and effort of analyzing and finding the most relevant attributes by using different feature extraction techniques or a mix of feature extraction approaches to obtain acceptable feature maps from the images dataset. The classification model's training and prediction times were dramatically lowered when CNN was used as a feature extractor. The model was able to achieve an accuracy of 99% with very few misclassifications in prediction. This proves that the use of CNN as a feature extractor and classifier is the optimal choice for the recognition of Assamese handwritten characters.

The results obtained here can be referred to in the future for the development of better feature extractor and handwritten character recognition.

CHAPTER-8

Future Directions

After studying and analyzing the presented Assamese handwritten character recognition system, here are future directions that could enhance the system:

1. Development of a well-labeled and vivid dataset of characters of Assamese language incorporating various writing styles from various individuals across the age groups and the communities. The mentioned act will ensure the data has sufficient variance in order to fit the model better.
2. Propose a novel pipeline with a hybrid feature extractor capable enough to detect different features and identify features with slightly different styles easily. The classifier can also be improved. Instead of softmax, the output can be passed on to a novel classifier with better classifying rates.
3. The recognition can be added with an NLP model, which will allow the recognition to spell the recognized text. The system will be helpful for knowing the right pronunciation and also will be aiding people with eyesight problems.

References

1. https://www.softwaretestinghelp.com/what-is-artificial-intelligence/

2. https://en.wikipedia.org/wiki/Optical_character_recognition

3. https://keras.io/examples/vision/handwriting_recognition/

4. https://en.wikipedia.org/wiki/Handwriting_recognition

5. Sueiras, Jorge. "Continuous Offline Handwriting Recognition using Deep Learning Models." *arXiv preprint arXiv:2112.13328* (2021).

6. https://en.wikipedia.org/wiki/Assamese_language

7. https://en.wikipedia.org/wiki/Bengali%E2%80%93Assamese_script

8.https://archive.ics.uci.edu/ml/datasets/Online+Handwritten+Assamese+Characters+Dataset#

9. Borah, Sagarika. "Segmentation of Assamese handwritten characters based on projection profiles." *International Journal of Computer Applications* 130.17 (2015).

10. R. R. Chowdhury, M. S. Hossain, R. ul Islam, K. Andersson and S. Hossain, "Bangla Handwritten Character Recognition using Convolutional Neural Network with Data Augmentation," 2019 Joint 8th International Conference on Informatics, Electronics & Vision (ICIEV) and 2019 3rd International Conference on Imaging, Vision & Pattern Recognition (icIVPR), 2019, pp. 318-323, doi: 10.1109/ICIEV.2019.8858545.

11. Nath, Keshab, and Subhash Basishtha. "Online and Offline Handwriting Recognition of Assamese Language—A Preliminary Idea." *Advances in Communication and Computing*. Springer, New Delhi, 2015. 183-192.

12. Das, Deepjoy, et al. "Performance comparison of online handwriting recognition system for assamese language based on HMM and SVM modelling." *International Journal of Computer Science Information Technology* 6.5 (2014): 87-95.

13. Bhattacharyya, Kaustubh, and Kandarpa Kumar Sarma. "ANN-based innovative segmentation method for handwritten text in assamese." *arXiv preprint arXiv:0911.0907* (2009).

14.Prasanna, S. R. M., et al. "Online Stroke and Akshara Recognition GUI in Assamese Language Using Hidden Markov Model." *arXiv preprint arXiv:1407.2390* (2014).

15.Pedamonti, Dabal. "Comparison of non-linear activation functions for deep neural networks on MNIST classification task." *arXiv preprint arXiv:1804.02763* (2018).

16.Schott, Lukas, et al. "Towards the first adversarially robust neural network model on MNIST." *arXiv preprint arXiv:1805.09190* (2018).

17. Baldominos, Alejandro, Yago Saez, and Pedro Isasi. "A survey of handwritten character recognition with mnist and emnist." *Applied Sciences* 9.15 (2019): 3169.

18. Dutta, Prarthana, and Naresh Babu Muppalaneni. "DigiNet: Prediction of Assamese handwritten digits using convolutional neural network." *Concurrency and Computation: Practice and Experience* 33.24 (2021): e6451.

.19.Singh, Harjeet, Rajendra Kumar Sharma, and V. P. Singh. "Online handwriting recognition systems for Indic and non-Indic scripts: a review." *Artificial Intelligence Review* 54.2 (2021): 1525-1579.

20. J. Sutha and N. Ramaraj, "Neural Network Based Offline Tamil Handwritten Character Recognition System," *International Conference on Computational Intelligence and Multimedia Applications (ICCIMA 2007)*, 2007, pp. 446-450, doi: 10.1109/ICCIMA.2007.86.

21. U. Baruah, S. M. Hazarika, "A Dataset of Online Handwritten Assamese Characters", Journal of Information Processing Systems, vol. 11, no. 3, pp. 325-341, 2015.

22. O'Shea, Keiron & Nash, Ryan. (2015). An Introduction to Convolutional Neural Networks. ArXiv e-prints.

23. V. Nair and G. E. Hinton, “Rectified linear units improve restricted boltzmann machines,” Haifa, 2010, pp. 807–814. [Online]. Available: https://dl.acm.org/citation.cfm

24. Y. LeCun, Y. Bengio, and G. Hinton, “Deep learning,” Nature, vol. 521, no. 7553, pp. 436–444, 2015. [Online]. Available: https: //doi.org/10.1038/nature14539

25. P. Ramachandran, B. Zoph, and Q. V. Le, “Searching for Activation Functions,” ArXiv, 2017. [Online]. Available: 1710.05941;http: //arxiv.org/abs/1710.05941

26. M. D. Zeiler, M. Ranzato, R. Monga, M. Mao, K. Yang, Q. V. Le, and G. E. Hinton, “On rectified linear units for speech processing,” in International Conference on Acoustics, Speech and Signal Processing. IEEE, 2013, pp. 3517–3521, IEEE

27. G. E. Dahl, T. N. Sainath, and G. E. Hinton, “Improving deep neural networks for LVCSR using rectified linear units and dropout,” in International Conference on Acoustics, Speech and Signal Processing. IEEE, 2013.

28. I. Goodfellow, Y. Bengio, and A. Courville, “Deep learning.” MIT Press, 2016. [Online]. Available: http://www.deeplearningbook.org

29. A. Krizhevsky, I. Sutskever, and G. E. Hinton, “Imagenet classification with deep convolutional neural networks,” 2012. [Online]. Available: http://citeseerx.ist.psu.edu/viewdoc/summary

30. V. Badrinarayanan, A. Kendall, and R. Cipolla, “SegNet: A Deep Convolutional Encoder-Decoder Architecture for Image Segmentation,” arXiv, 2015. [Online]. Available: http://arxiv.org/abs/1511.00561

31. M. Lin, Q. Chen, and S. Yan, “Network In Network,” arXiv, 2013. [Online]. Available: http://arxiv.org/abs/1312.4400

32. https://pandas.pydata.org/

33. https://numpy.org/

34. https://scikit-learn.org/stable/index.html

35. https://www.tensorflow.org/api_docs/python/tf

36. https://keras.io/guides/

37. https://matplotlib.org/

38. https://seaborn.pydata.org/

39. https://opencv.org/

40. Sarma, Kandarpa Kumar. "MLP-based Assamese Character and Numeral Recognition using an Innovative Hybrid Feature Set." *IICAI*. 2007.

41. Bania, Rubul Kumar, and R. Khan. "Handwritten Assamese character recognition using texture and diagonal orientation features with artificial neural network." *Int J Appl Eng Res* 13.10 (2018): 7797-7805.

42. Kussul, Ernst, and Tatiana Baidyk. "Improved method of handwritten digit recognition tested on MNIST database." *Image and Vision Computing* 22.12 (2004): 971-981.

43. Alvear-Sandoval, Ricardo F., José L. Sancho-Gómez, and Aníbal R. Figueiras-Vidal. "On improving CNNs performance: The case of MNIST." *Information Fusion* 52 (2019): 106-109.

44. B. Sarma, K. Mehrotra, R. Krishna Naik, S. R. M. Prasanna, S. Belhe and C. Mahanta, "Handwritten Assamese numeral recognizer using HMM & SVM classifiers," *2013 National Conference on Communications (NCC)*, 2013, pp. 1-5, doi: 10.1109/NCC.2013.6488009.

45. Jain, Ubeeka, and D. Sharma. "Recognition of Isolated Handwritten Characters of Gurumukhi Script using Neocognitron." *International Journal of Computer Applications* 10.8 (2010).

46. Medhi, Kalyanbrat, and Sanjib Kr Kalita. "Assamese digit recognition with feed forward neural network." *International Journal of Computer Applications* 109.1 (2015).

47. Ishak, Wan Hussain Wan and Shuzlina Abdul Rahman. “Neural network in handwritten recognition system: A survey.” (2003).

48. Sinwar, Deepak, et al. "Offline script recognition from handwritten and printed multilingual documents: a survey." *International Journal on Document Analysis and Recognition (IJDAR)* 24.1 (2021): 97-121.