Classification for Low Intra-Class Variability and High Number of Classes

By

Mahefroze Shaheen

Fall 2018-MS (CS-08)-00000273614

Supervisor:

Dr. Khawar Khurshid

DEPARTMENT OF COMPUTER SCIENCE

SCHOOL OF ELECTRICAL ENGINEERING & COMPUTER SCIENCE

NATIONAL UNIVERSITY OF SCIENCES AND TECHNOLOGY (NUST),

ISLAMABAD, PAKISTAN.

March 2022

Online Printing Date & Time: Tuesday, 19 July 2022 22:54:40

THESIS ACCEPTANCE CERTIFICATE

Certified that final copy of MS/MPhil thesis entitled "Classification for low intra-class variability and high number of classes" written by MAHEFROZE SHAHEEN, (Registration No 00000273614), of SEECS has been vetted by the undersigned, found complete in all respects as per NUST Statutes/Regulations, is free of plagiarism, errors and mistakes and is accepted as partial fulfillment for award of MS/M Phil degree. It is further certified that necessary amendments as pointed out by GEC members of the scholar have also been incorporated in the said thesis.

Signature: [illegible]

Name of Advisor: Dr. Khawar Khurshid

Date: 07-Jul-2022

HoD/Associate Dean: ____________________

Date: ____________________

Signature (Dean/Principal): ____________________

Date: ____________________

Publish Date & Time: Thursday, 14 July 2022 , 12:22:32

Online Printing Date & Time: Tuesday, 19 July 2022 22:55:03

Approval

It is certified that the contents and form of the thesis entitled "Classification for low intra-class variability and high number of classes" submitted by MAHEFROZE SHAHEEN have been found satisfactory for the requirement of the degree

Advisor : Dr. Khawar Khurshid

Signature: [illegible]

Date: 07-Jul-2022

Committee Member 1: Mr. Muhammad Imran Abeel

Signature: [illegible]

Date: 14-Jul-2022

Committee Member 2: Dr. Salman Abdul Ghafoor

Signature: Salman

Date: 07-Jul-2022

Committee Member 3: Dr. Ahmad Salman

Signature: [illegible]

Date: 14-Jul-2022

Publish Date & Time: Thursday, 14 July 2022 , 12:22:32

Dedication

I dedicate my thesis to my beloved parents and my husband who had been a source of constant support, encouragement and motivation. I am truly thankful to all these people in my life

Online Printing Date & Time: Tuesday, 19 July 2022 22:55:44

Certificate of Originality

I hereby declare that this submission titled "Classification for low intra-class variability and high number of classes" is my own work. To the best of my knowledge it contains no materials previously published or written by another person, nor material which to a substantial extent has been accepted for the award of any degree or diploma at NUST SEECS or at any other educational institute, except where due acknowledgement has been made in the thesis. Any contribution made to the research by others, with whom I have worked at NUST SEECS or elsewhere, is explicitly acknowledged in the thesis. I also declare that the intellectual content of this thesis is the product of my own work, except for the assistance from others in the project's design and conception or in style, presentation and linguistics, which has been acknowledged. I also verified the originality of contents through plagiarism software.

Student Name: MAHEFROZE SHAHEEN

Student Signature: [illegible]

Publish Date & Time: Thursday, 14 July 2022 , 12:22:32

ACKNOWLEDGEMENT

I am thankful to Allah Almighty for His endless blessing over me because of which I have been able to complete my thesis. I am thankful to Allah, for His guidance at every step during my research and making me move forward in carrying out and completing it.

Special thanks to my supervisor Dr. Khawar Khurshid for his guidance. This research would not have been possible without his guidance and advice. Through his guidance I had been able to overcome the obstacles and difficulties I faced during my research. I would like to thank Sir and his student Ms. Benish Aman for providing me with the CEFAR Dataset for carrying out my research.

I am extremely thankful to my parents for their constant support and encouragement throughout my entire master's studies. I am truly humbled and blessed for their love and support.

I owe a special thanks, to a very important person in my life, my husband who supported me throughout my thesis. He not only motivated me but also took care of our daughter during my absence and managed both work and home. I really appreciate how my beloved daughter Hania patiently bore my absence.

Mahefroze Shaheen

Contents

List of Figures

List of Tables

ABSTRACT

Optical Character Recognition (OCR) is the recognition of handwritten or printed text for various digital processing tasks. It is an important area of research in the field of image processing, natural language processing and artificial intelligence. Many real world applications, for example price tag scanners, online translators and text to speech converters are based on OCRs. The research and work for English OCR has been quite remarkable. Many robust OCR systems have been developed for English language. The research on Urdu OCR is quite recent and till date, no sophisticated Urdu OCR system exists. Urdu is the national language of Pakistan and is spoken by over 300 million people around the world. Owing to this importance, there is a need to develop a method for recognition of Urdu script. The lack of attention to Urdu OCR is due to the complexity of this language. It is a highly cursive and context sensitive language. A single word has numerous structural variations, which make it difficult to be recognized. Due to these challenges, no benchmark dataset for Urdu could be developed. The focus of our research is to develop an effective method for recognition of Urdu script. In our proposed method, we developed an Urdu ligature dataset named CEFAR dataset and used deep learning to recognize these ligatures. The dataset contained exhaustive combinations of Urdu characters of length 2 and 3. These ligatures were in the form of images divided into 3 parts; 2 and 3 character ligature sets separately and the third part contained ligatures of both 2 and 3 characters. The aim was to train a deep learning model for recognizing these ligatures. The main challenge associated with this data was its high number of classes and low intra-class variability. In our proposed method, we developed a novel technique to solve this problem by using data augmentation to increase the number of representative samples within each ligature class and then applied class redefinition for class reduction. Data augmentation was followed by ligature recognition using Recurrent Neural Networks. RNNs were employed for classification of these ligatures. A 34 layer recurrent neural network model was used with a final fully connected layer for classification. Three separate models were trained and evaluated for the three ligature sets. All the three ligature's training models gave remarkable performance. The 2 character ligature set gave an accuracy of 96.4%, 99.7% accuracy on three character ligature set and 97.5% accuracy on the combined ligature set of 2 and 3 characters. The overall performance considered for the recognition system was the accuracy of the model over the combined 2 and 3 character ligature dataset, which was 97.5%. Our model performed brilliantly well than the existing deep learning methods for Urdu ligature recognition. The excellent classification accuracy of our deep learning model makes this research play an effective role in not only building a benchmark Urdu dataset but also its classification. In future, this research could be extended for development of classification systems for ligatures of lengths greater than three.

Chapter 1

Introduction

1.1 Motivation

Image classification involves labelling or categorization of images based on their similarity. Image data is increasing every day which makes image classification a very important aspect in the field of computer vision and data science [1]. It has wide range of applications in various fields like weather forecasting, object detection, medical diagnosis, character, and pattern recognition etc. [2]. The amount of image data is being produced at such a rapid pace that it requires an efficient method for classification. The commonly used methods are machine learning and deep learning [3]. Machine learning in simple terms is making the machines capable of human understanding. Many algorithms are present that make the machine learn patterns. for example, Neural Networks, SVM, Naïve Bayes etc. Deep learning is a type of machine learning which extracts information regarding the patterns directly from the data. Some of the deep learning algorithms are LSTM, deep Neural Networks, and auto encoders etc. [3].

Performance and efficiency of the classification algorithms highly depend upon the features learned from the images. Machine learning algorithm involves feature engineering method. On the other hand, deep learning uses feature extraction method which extracts the features of an image directly from the raw pixels. This method of feature extraction makes deep learning a better approach for image classification. Moreover, the large amount of image data present also proves deep learning to have shown excellent performance for image recognition tasks [3]. Optical Character Recognition (OCR) is a very important area of research in pattern recognition and Artificial Intelligence. It is basically the conversion of handwritten or scanned documents into machine readable format which are processed later [7]. The digital image documents are processed further to perform various tasks such as text to speech recognition, document analysis and pattern matching. Many real-world applications have been built using the OCR like fingerprint scanners, price tag scanners, online translators on web or smart phones. Adobe and Google Drive are one of the examples of these OCR systems that have commercial importance [3].

The basic task that an OCR performs is the recognition or classification of the text of the language for which it is made. The workflow of an OCR system has six phases, namely Image Acquisition, Pre-processing, Segmentation, Feature Extraction, Classification / Recognition, post processing. The first step of image acquisition is the digitization of the textual document. The acquired image is then pre-processed by performing image processing. Segmentation is highly based on type of OCRs. In some OCRs the image containing the text is segmented at character or ligature level and some use the holistic approach that aims at the recognition of data without segmentation. These are all the pre-liminary steps towards the main task performed by the OCR i.e., the Classification of the text. This involves the use of machine learning or deep learning approach. In most of the recent OCR systems the classification task is done using the deep learning models as they perform well. Post processing is an optional step and may not be required in some OCR systems [3].

So, the focus of an OCR is the classification of the text inside the image. It involves image processing, computer vision and pattern recognition. Making OCR systems highly significant to these fields.

The history of OCR systems is quite old which dates to the early 1950's. OCR system have been developed for many languages such as English, Arabic, Chinese, Latin, Japanese and many other [6]. One common aspect of these language is that the text of these languages is non-cursive and has separate characters which limits the capability of these OCR systems. Little work has been carried on the languages having cursive script like Arabic and Urdu.

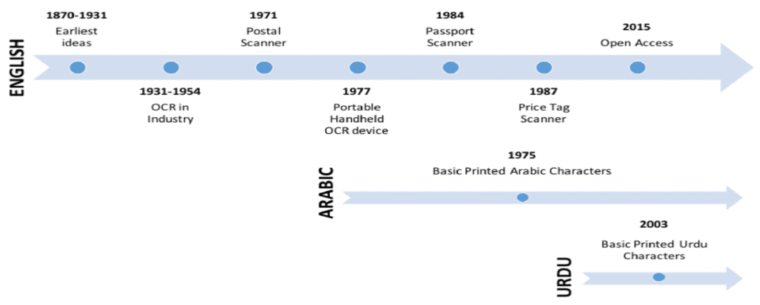

ENGLISH
1870-1931 Earliest ideas
1931-1954 OCR in Industry
1971 Postal Scanner
1977 Portable Handheld OCR device
1984 Passport Scanner
1987 Price Tag Scanner
2015 Open Access
ARABIC
1975 Basic Printed Arabic Characters
URDU
2003 Basic Printed Urdu Characters

Figure 1.1: Timeline comparison of English, Arabic and Urdu OCR

Work on cursive scripts like Arabic and Urdu is quite recent. Significant attention has not been given to the recognition of these languages due to the complexity of their scripts [3]. The work on Arabic language OCR started in the early 1970s and rapidly grew to better Arabic OCR systems. Comparing English and Arabic OCR systems to that of Urdu shows that the work on Urdu language is very recent with only separate characters being recognized with a high accuracy. Urdu language has quite a lot of similarity to Arabic language. But the work on Arabic OCR has advanced more than Urdu. The reason being the writing style of the Urdu Language. Arabic Language is written in Naskh writing style while Urdu follows the Nastalique style of writing, which is highly cursive and context sensitive. This difference makes Urdu a more complex language in case of OCRs. The pattern recognition techniques used for Arabic can not be applied on Urdu hence causing a lag in its recognition. There is a huge gap on the research of Urdu language OCRs which needs to be addressed [3].

Urdu is the national language of Pakistan and is spoken and understood in many parts of the world including Pakistan, India, and Bangladesh. It is an Indo-Aryan Language that originated from a combination of other languages i.e., Arabic, Pushto, and Persian. It is spoken worldwide by more than 300 million people [3][6][9].

Considering the importance of Urdu language. It is important that significant research be carried out in the field of pattern recognition aiming to make its advancement in the field of Artificial Intelligence. This motivates my research in the field of pattern recognition in the Urdu Language. The focus of the research is the classification of Urdu text using image processing and deep learning. We have developed a classification system for Urdu text using deep learning algorithm that performs well than the existing Urdu classification models.

1.2 Research Problem

1.2.1 Types of OCR systems:

OCR systems can be divided into various types based on.

Input Method:

OCR systems are categorized into Online and Offline. In an Online OCR system, the data is acquired in real time and is recognized while in the offline OCR that text comes int the form of image already acquired from a document [3].

Writing mode:

The text to be recognized can be handwritten or printed/digital format. Handwritten OCR is highly complex as it is very difficult to recognize the handwritten text because of varying writing style for different people [3]. On the other hand,

printed text is always offline and poses other challenges like image quality and writing styles.
Font (Single font / Omni font):
OCR systems can have the capacity to recognize language in single font or multiple forms. This categorization is important as OCRs can work properly in on font while they may completely fail to recognize in different font style.
Script Connectivity (Isolated / Cursive):
An isolated script of a language does not have connected characters, rather the characters do not join to each other. A cursive script has characters joined to each other. A cursive script makes the task of recognition a difficult process.
Most of the research carried on text recognition systems for Urdu language so far are offline and use printed documents that use single font. The font style used in Urdu is the Nastalique font that is highly cursive. Because of this highly cursive nature of Urdu language, most of the classification is performed on separate Urdu characters while classification at ligature (combination of one or more character) or word level is still not significant [3][7].

1.2.2 Working of OCR

The flow of work in an OCR system consists of the following steps:

1. Image Acquisition
2. Pre- Processing
3. Segmentation
4. Feature Extraction
5. Classification
6. Post Processing

Image acquisition for offline OCR is always in the printed form. This could be an image acquired through a scanner or digitally produced textual image. The image after acquisition needs to be pre-processed for removing noise, distortions, or clutter. Pre-processing is an important step as noise in the image can affect the quality of the classification model to recognize the image. Once the image is pre-processed, the next step of segmentation is highly variable. Once the image containing the text has been acquired, there is a need to isolate the text from it. There are various levels of segmentation these are at page, line and text level [3].

Once the text has been acquired, then there are two very important and prominent approaches used for segmentation:

1. Analytical Approach
2. Holistic Approach

Analytical method segments the text into isolated characters for recognition while the holistic method recognizes the text at word or ligature level. The drawback of analytical approach of text segmentation is that it becomes highly computationally intensive because of numerous characters acquired after segmentation. Holistic approach on the other hand recognizes a word or ligature, covering up the computationally intensive nature of the analytical approach [3].

Most of the research focus for Urdu language has been shifted towards the holistic method of text segmentation as it has proved to provide better recognition results [3][6].

Once the text has been segmented the next and the major task is the classification of this text. Classification either used machine learning or deep learning for developing a system capable of recognizing the language for which the OCR is built. Research has shown machine learning models to perform well for less amount of data but when the dataset increases, machine learning cannot give satisfactory results for the image classification. Deep learning has shown remarkable results in the classification / recognition tasks because it can handle large volume of data. Same goes for the Urdu classification systems, where deep learning has outperformed machine learning and is the recent focus of the studies [3].

1.2.3 Performance Criteria of OCR

The efficiency with which an OCR system recognizes the text depends on the dataset, quality of the feature extracted from the dataset and the classification method used to classify the images [3]. All these factors play a vital role in the accuracy with which a classification model recognizes the text in an OCR system [3].

1.3 Urdu OCR

1.3.1 Origin and Background of Urdu Language

The word Urdu originates from Turkish word “Ordu” meaning “army’ or “clan”. It is influenced by other languages such as Arabic, Turkish and Pashto. It is a rich language that flourished mainly in the Mughal Era [3].

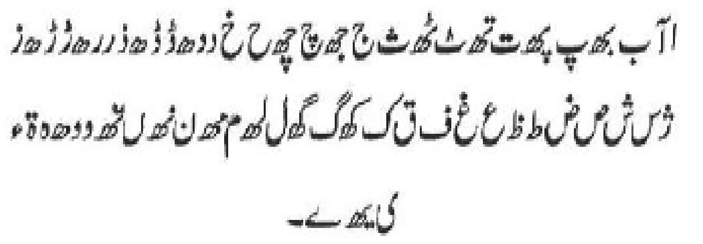
Urdu language has writing style similar to Arabic and Persian because of which it is written in Perso-Arabic script. It follows the Nastalique style of writing [3]. Nastalique is a combination of two writing scripts i.e. Nash and Taliq and was later called Nastalique. This script emerged and prospered in the times of Mughal emperors and therefore engraved in the parts of Pakistan, India and Bangladesh

[3]. Urdu became the national language of Pakistan after its independence and therefore has a very important place. It is spoken, written and understood all over Pakistan, India as well as Bangladesh [7].

1.3.2 Characteristics of Urdu Language

The Urdu script has total of 58 alphabets which include;

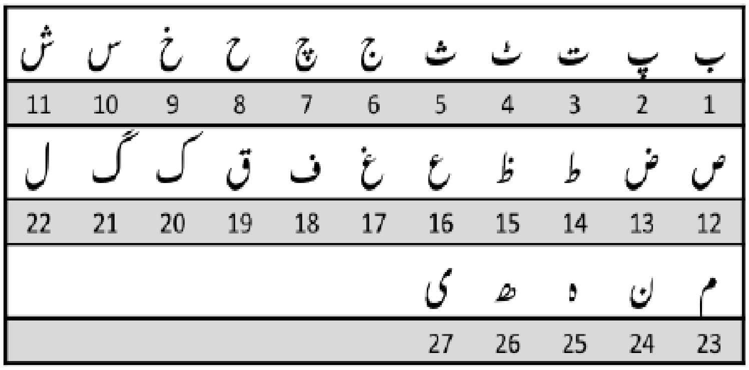
• 38 primary characters: The basic alphabets that can be classified into 10 classes by combining alphabets with similar base form.
• 15 Diacritic marks
• Numerals
Nastalique is a diagonal script with characters read from left to right and text lines



ا آ ب بھ پ پھ ت تھ ٹ ٹھ ث ج جھ چ چھ ح خ د دھ ڈ ڈھ ذ ر رھ ڑ ڑھ ز
ژ س ش ص ض ط ظ ع غ ف ق ک کھ گ گھ ل لھ م مھ ن نھ و وھ ہ ھ ء ۃ
ی یھ ے ۔

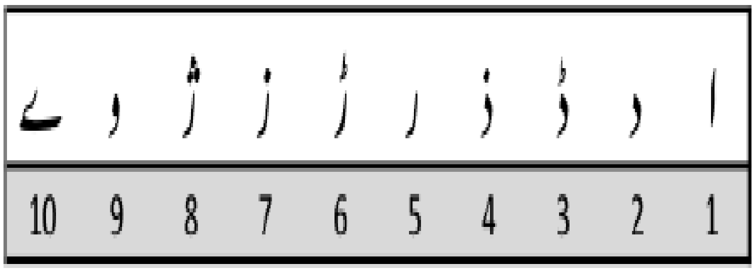
Figure 1.2: Basic Urdu Characters

read from top to bottom [3].
Joiner and Non-Joiner characters:
The primary characters are grouped based on their joining capacity to other characters. The characters which change their shape based on the neighboring connecting character or its position within the character are called joiner characters. All joiner characters can have four forms which is isolated, start, middle and end. The non-joiners only have a single shape and remain either isolated or come at the end. There are 27 joiner characters and 10 non joiners. Figure 1.3 and 1.4. represent the joiner and non-joiner Urdu characters [3].



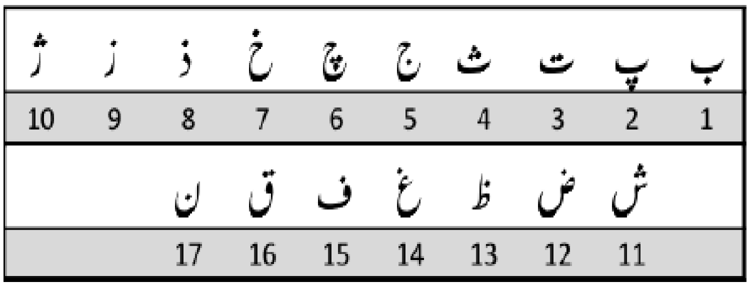
ش	س	خ	ح	چ	ج	ث	ٹ	ت	پ	ب
11	10	9	8	7	6	5	4	3	2	1
ل	گ	ک	ق	ف	غ	ع	ظ	ط	ض	ص
22	21	20	19	18	17	16	15	14	13	12
						ی	ھ	ہ	ن	م
						27	26	25	24	23

Figure 1.3: Joiner Urdu Characters



ے	و	ژ	ز	ڑ	ر	ذ	ڈ	د	ا
10	9	8	7	6	5	4	3	2	1

Figure 1.4: Non-joiner Urdu Characters



ژ	ز	ذ	خ	چ	ج	ث	ت	پ	ب
10	9	8	7	6	5	4	3	2	1
		ن	ق	ف	غ	ظ	ض	ش	
		17	16	15	14	13	12	11	

Figure 1.5: Urdu character containing dots

Dots and Diacritics: These are the special marks that lie above or below the main body of the Urdu characters. These dots and diacritic marks let the characters have different pronunciations [3]. They are of three types:

•**Dots:** They lie above or below the main body of the character [3]. 17 Urdu characters have dots placed on them and are shown in the Figure 1.5.

•Aerab:
These are special marks that lie above or below the character acts as vowels and let a single character have multiple sounds [3].

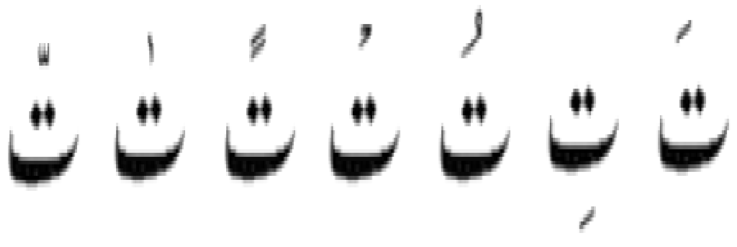


Figure 1.6: Special marks in Urdu shown over a single character

• Retroflex Consonants:
Three characters in Urdu language have a superscript ط placed on them and are spoken with the tongue having a flat, curved or concave shape [3]. These characters are shown in the Figure 1.7.

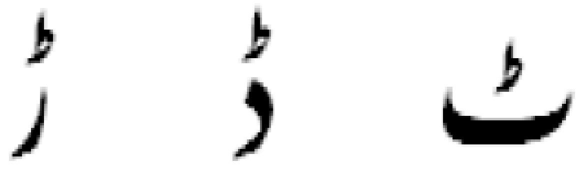


Figure 1.7: Urdu characters showing retroflex constant

1.3.3 Challenges of Urdu Script

Arabic and Urdu languages are written in various calligraphic styles. Some of the prominent calligraphic styles include Natsalique, Koufi, Diwani, Naskh and Rouqi. The calligraphic style adopted for Urdu writing is Nastalique. Nastalique is a highly cursive and context sensitive script. The complexity of this writing style has caused a huge issue for its recognition in an OCR system. Following are some of the main challenges related to the Urdu recognition [37].

Diagonality: The Nastalique Urdu script is diagonal in nature meaning the characters are written from top left to bottom right. All the words are tilted at a certain angle. Not only this but each character is tilted at its own random angle, making the recognition or segmentation of this script extremely difficult. This diagonality though reduces the space for text writing but it causes overlap among the ligatures when written in text lines. A ligature can overlap with another ligature written at the bottom [37].

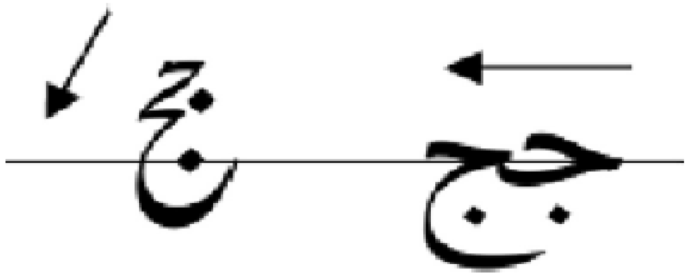


Figure 1.8: Figure showing diagonal nature of Nastalique script

Bidirectional Nature:
The script is bidirectional. The words are written from left to right whereas the numerals are written from right to left. Figure 1.9 demonstrates this bidirectional nature of Urdu script [37].

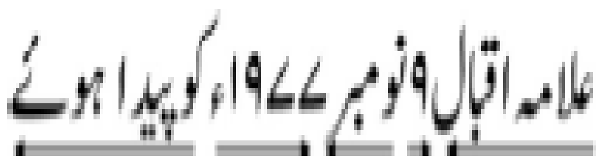
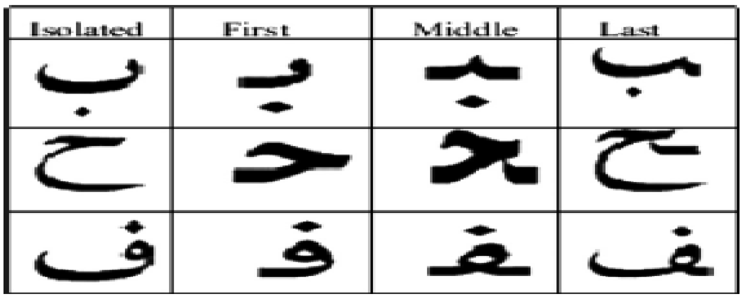


Figure 1.9: Bidirectional nature of Nastalique script

Cursive:
Nastalique script is highly cursive, characters in a ligature are connected to each other at specific points. The various connective points of Urdu language make it highly cursive and complex for recognition [37].
Context Sensitivity:
Characters in Urdu language are context sensitive in nature. A single word can

have various shapes based on the context in which it is present meaning the character can have multiple different shapes in the presence of a different neighboring character. A single character can change its shape based on its position within the ligature. A single character can have 4 forms i.e., isolated, start, middle and end [37]. Figure 1.10 shows the various shapes of few characters.



Isolated	First	Middle	Last
ﺏ	ﺑ	ﺒ	ﺐ
ﺡ	ﺣ	ﺤ	ﺢ
ﻑ	ﻓ	ﻔ	ﻒ

Figure 1.10: Shape variations of few Urdu characters

Overlapping:
The ligatures can overlap with other ligatures within the script. The overlapping is vertical meaning that the ligature from one line can overlap vertically with another ligature present in the bottom line of the other ligature but does not touch it. This is done to save the white space but on the other hand creates a challenge during the recognition phase [37]. There can also be inter and intra ligature overlapping.

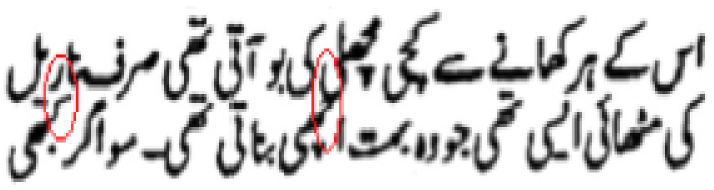


Figure 1.11: Vertical overlapping between ligatures

The highlighted portions in the figure represent the ligature overlapping.

Upper and Lower-case letters:
There is no differentiation as to upper and lower case letters in Urdu as present in other languages like English. Only the last character is present in its full form in a word or ligature or is therefore considered as upper case [37].
Placement of Dots:

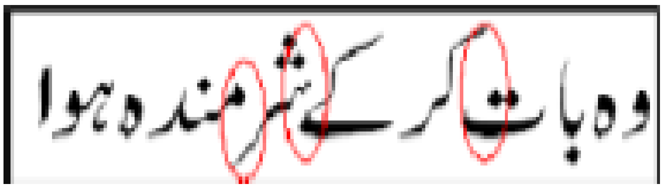

وہ بات کرکے شرمندہ ہوا

Figure 1.12: Ligature overlapping within script(example 1)

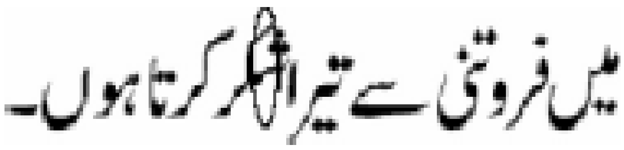


Figure 1.13: Ligature overlapping within script(example 2)

In Urdu writing characters have dots placed above or below the body of the character. But there is no standard rule of the placement of dot. In order to reduce the extra space or due to context sensitivity or overlapping, the dots can be places at a different position leading to ambiguity of the correct position of the dot. This becomes highly difficult for the recognition system to identify the correct position of this dot and therefore the incorrect classification [37].

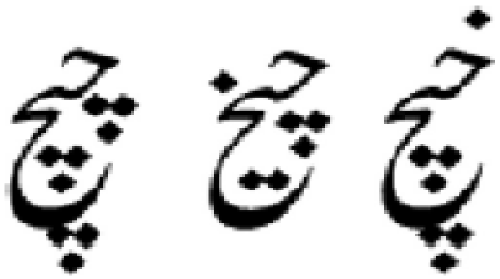


Figure 1.14: Dots placement in Urdu ligatures

Stretching:
Some characters can also undergo stretching while writing in order to reduce the writing space. Stretching means the shape of the character is elongated or changed then the normal shape of the character. Few characters only change their width while their shape remains the same [37].

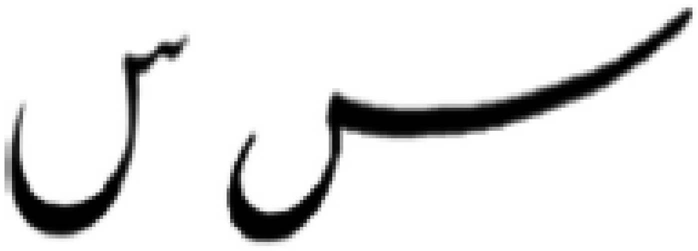


Figure 1.15: Ligature Stretching

Positioning:
It is a way to write Urdu script. Mostly to accommodate long words in Urdu language the position of ligatures in changed. A ligature can be placed at the top of the previous ligature. This at on hand reduces the space but varies the shape of ligatures within words and can cause confusion in the recognition [37].

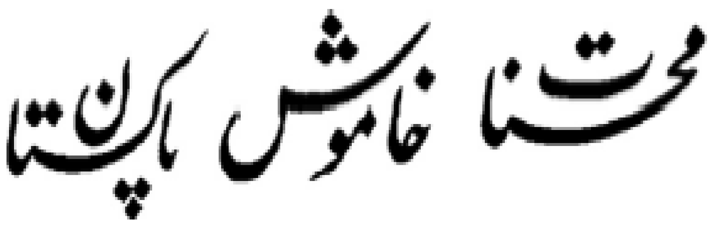


Figure 1.16: Variable ligature positioning within words

These are some of the complexities of Urdu scripts that cause difficulty in the recognition of Urdu language. Besides the absence of benchmark data, these variations in the writing style cause lower recognition rates of the recognition systems. A lot of attempts have been made to cater some of these challenges but still not all of them could be resolved. Resolution of these issues are still under investigation [3][37].

1.3.4 Classification in Urdu Classification Systems

The basic working of OCR and performance criteria has been explained. The focus of our study is the classification step for the Urdu OCR system. Classification is the main focus in an OCR system as the previous steps of image acquisition, preprocessing and segmentation are preliminary steps for classification [3][7].

Classification of English and Arabic text has been already carried out with remarkable achievements. But the work on Urdu text classification is still way behind. The reason for less work performed for Urdu classification systems is the cursive nature of this language and the lack of benchmark dataset. Cursive means the script connectivity, being highly cursive, the characters are joined to each other which lead to difficulty in the segmentation of the text. When a cursive text is segmented it can create error in the word or ligature being recognized and it also becomes computationally intensive. The other major reason for gap in research on Urdu classification research is the absence of a benchmark dataset [3][7][11].

Benchmark dataset is very important for the evaluation of a system. Few synthetic datasets, are available, these are:
CLE: Contains ligatures from various categories like sports, games, news, finance, culture, and entertainment [3].
UPTI (Urdu Printed Text Image Dataset): It contains 10063 text and ligature images [3].
UPTI 2.0: Contains 18000 ligatures covering 70% of the practically occurring ligatures [3].
Classification has been carried out on them. But the results of classification on these datasets are not very remarkable as the size of the datasets is small and they do not cover all the possible ligatures [3].

1.3.5 Challenges in Urdu Language Classification:

Classification using deep learning requires a large amount of training data for providing better results [10]. The available Urdu Language datasets do not have large amount of data needed for training the classification models. Moreover, they do not cover all the possible ligatures in the Urdu language leading to scarcity and inefficiency of the recognition systems.

In order to have better recognition systems for Urdu language there is a need to have large dataset that covers all ligatures providing better recognition results.

1.4 Problem Statement

A large dataset of Urdu ligatures has been developed that contains exhaustive combination of 2 and 3 Urdu characters comprising of 454720 training images and 90944 test images in binary format. The uniqueness of this dataset from all the other available datasets is its exhaustive nature. While other datasets contain the most commonly occurring ligatures, this dataset has exhaustive combination covering up all the possible ligatures of the Urdu language.

Classification of ligatures present in this dataset needs to be carried out for developing a better recognition system. The classification accuracy using deep learning highly depends on the size of the dataset and the number of images within each category. This dataset though large enough poses the class imbalance problem which is a serious classification challenge and can affect the classification accuracy of the recognition system [10]. This dataset contains 32480 classes. Such a high number of classes can cause lower classification accuracy of the training model as well as can be highly computationally intensive.

Urdu recognition systems lack large datasets. The available datasets do not cover all the possible ligatures. Classification systems built on these datasets cannot perform well [11]. This thesis aims to propose a method to solve the problem of classification for Urdu dataset containing large number of classes with low intra-class variability.

1.5 Research Objective

The objective of this study is.

• To develop a method for reducing the low intra-class variation within the dataset containing large number of classes.

• Use deep learning approach to classify the dataset for building a recognition system with higher classification accuracy than the already available classification models.

• Analytical and statistical evaluation of the results of the classification model.

• Comparison of the proposed study to previous research carried out.

1.6 Significance

OCR holds a very important research place in the fields of pattern recognition, image processing, document analysis and Artificial Intelligence. They hold commercial importance as they are used in a wide range of applications in the everyday life. Being language specific, they can fulfill the needs in various applications used by people belonging to different parts of the world. This research is focused on the recognition of the Urdu text in images. This study would be significant in the following ways;

1.6.1 Commercial Applications

Work in the field of Urdu OCR is quite recent. Research is being carried out on the recognition of text in the Urdu digital documents but there are still many gaps present. Most of the natives of Pakistan can only read, write, and speak in Urdu language. As the technology is advancing everywhere there is a need of processing of Urdu language in the modern world technologies such as smart phones and tablets. This research work can be applied to the document processing for natives of Pakistan such as legal documents, newspapers, magazines, academic, religious and poetic books. It can help in building many commercial applications such as handheld devices for text scanning, document searching and auto translation for handicapped people. Digital dictionaries, data entry and processing softwares for Urdu language can be built. [12]

1.6.2 Better Performance Recognition System

The prominent existing datasets for printed Urdu text include CLE, UPTI, UPTI 2.0. Others include many custom datasets built for Urdu text recognition research such as developed by [13][14][15]. Work on Urdu language has gained serious attention only in the past 10 years [7]. Due to its recent prominence and understanding of Urdu digitization, the need for a benchmark dataset has emerged. We have developed a dataset of exhaustive Urdu ligatures in printed format. The feature of this dataset is its large number of ligature images and the ligature coverage. This makes it unique to the existing datasets. These are the prominent features required by any deep learning model for model training. The aim of this study is to use deep learning to classify these ligature images with better classification accuracy than the existing classification methods.

1.6.3 Handle Low Intra-Class Variability and Large Number of Classes

The exhaustive dataset at hand is quite large which may help in getting a better recognition model than the existing ones. But there is a serious challenge associated with this dataset which is class imbalance. Class imbalance can lead to lower classification accuracy [10]. In our research work we have also handled this issue to train a better classification model.

1.7 Structure

This thesis is organized into five chapters.
Chapter 1- Introduction, the first chapter introduces the topic of research and highlights the problem statement and scope of research
Chapter 2- Literature Review, it is a detail of the research carried out in the past years regarding the Urdu recognition Techniques. It will discuss in detail the existing Urdu datasets, methods, techniques for Urdu classification and the classification accuracies of the previous research work.
Chapter 3- Methodology, this section is a detailed explanation of the work carried out in our research work. It starts from the initial phase till the final including explanation of pre-processing of the dataset, the feature extraction, training methodology and the how the final classification is performed.
Chapter 4- Results, as the name suggests, this section of this thesis explains the outcomes of the experiments done over the dataset. It details the training phase results, fine tuning outcomes and the final classification accuracy obtained over the dataset.
Chapter 5- Evaluation and Discussion, this chapter is the evaluation of the results obtained. It details the previous work and then explains the outcomes of our research along with its evaluation and interpretation. It analyzes the research work in depth. In the end, it also explains the limitation in our research and states the future directions for further research.
Chapter 6-Conclusion,this chapter summarizes the whole thesis. Describes the problem statement and research objectives along with the contribution of this thesis, its limitations and future directions.

Chapter 2

Literature Review

Recognition Systems make up a very prominent part in field of image processing, document analysis and pattern recognition [2][3][7]. These recognition systems have a long history in case of English Language and many mature and sophisticated English recognition system have been developed [3]. The need for Urdu character recognition has emerged because of the widespread use and understanding of this language by more than 300 million people [6][9]. It is ranked as the fifth most spoken language of the world with 4.7% of the total world population [15].

The research on Urdu language recognition system has attracted a lot of attention in the recent years. Many attempts have been made for developing better recognition systems [3].

OCR is a complete process that processes the raw printed document to its final recognition. Research for Urdu language OCR is ongoing. Researchers have been able to recognize separate Urdu characters with high accuracy but in case of joined characters or ligatures the progress is way far behind [7]. This chapter explains in detail the work done by previous researchers on Urdu OCR. It explains the different types of Urdu datasets, steps of the OCR systems and the work done in each of these steps for Urdu script. It also discusses primarily the classification datasets and methods reported in the literature.

2.1 Urdu Datasets

Deep Learning algorithms work on large and complex datasets [3][1]. There have been no remarkable achievements obtained on the research of Urdu OCR as compared to the OCR for other languages including English and Arabic. Research on these languages has developed quite mature OCRs with a very high recognition rate. The less growth of Urdu language in recognition systems is not only because of the complexity, context sensitivity and cursive nature of this script but a big reason for it is the absence of a benchmark dataset. There is no benchmark dataset for Urdu language. It is very important in any deep learning study to have a ground truth available that can be used for the evaluation of the dataset. This gap has resulted in lack of growth in the field of OCR for Urdu [1][3][11].

Over the time attempts have been made to build datasets for Urdu language that could be used as standard. Figure 3 represents the various Urdu datasets.

Some of the Urdu datasets built up so far are as follows:

2.1.1 EMILLE (Enabling Minority Language Engineering) project:

EMIILLE was a project initiated by University of Lancaster and Sheffield in 2003. The objective of this project was to create a dataset for South Asian Language to be publicly available. It has three types of data i.e. monolingual, annotated and parallel. It is a huge dataset of 96 million words including 512000 spoken Urdu words and 1640000 collected from Urdu text [5].

2.1.2 Corpora and associated tools for Urdu text processing by Centre of Language Engineering (CLE):

CLE is working in Pakistan with an aim to make regional languages available to local people through communication and technology. A large corpus has been developed having 19.3 million Urdu ligatures. These ligatures are collected from various domains of life such as sports, games, finance, culture, entertainment, consumer information and personal information [14].

2.1.3 Urdu-Jang Dataset:

Image Understanding and Pattern Recognition Group at the Technical University of Kaiserslautern, Germany has produced a synthetic dataset of Urdu language from the famous Pakistani newspaper named Jang. This dataset contains 26925 UTF encoded Urdu text lines [8].

2.1.4 Urdu Printed Text Image Database (UPTI):

This an Urdu dataset having 10,063 synthetically generated images of text lines and ligatures. It has images for both line and ligatures versions. Various data degradation procedures have been applied to increase the size of this dataset. Techniques of jitter, threshold, sensitivity and elongation have been applied on 12 sets of data. This dataset has words and ligatures collected from the Jang newspaper containing words from different aspects. These include words from social, political and religious issues [16][3].

2.1.5 Urdu Nastalique Handwritten Dataset (UNHD):

It is a handwritten dataset of Urdu having some of the commonly used ligatures and numerals. This dataset was collected from 500 writers including both male and female on A4 size paper. It has 312000 words and 10000 text lines. The data collected on paper was scanned and has word written in the nastalique font [16].

2.1.6 Other Datasets:

Other small datasets have also been generated in an attempt to create Urdu benchmark datasets. These include;
• A dataset of 25 documents having ground truth for OCR and layout analysis [46].
• A dataset developed by [13] 50,000 words collected from various fields such as sports, games, news, social communication, entertainment and consumer information. This dataset is known as Ijaz and Hussain dataset.
• A 10,000 words dataset collected from various domains naming press, novels, letters, religion, culture, healthcare, sports, stories, science, translations and book reviews [14].

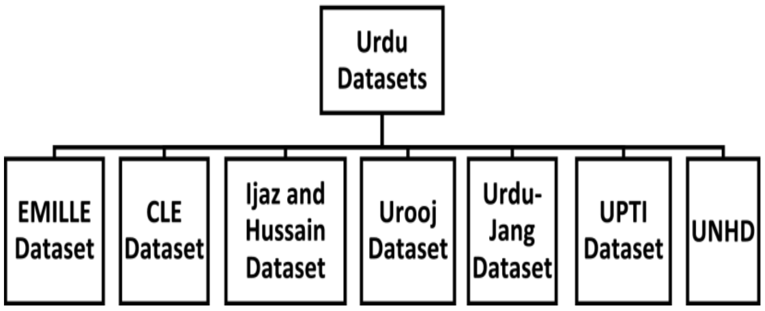


Figure 2.1: Urdu Datasets reported in literature

2.2 Categories of OCR

OCRs are divided into different types based on their features. Following are the different criteria which define the type of the OCR system.

2.2.1 Input Method:

The input to the recognition system is a text image but the way it is obtained and processed by the OCR system, divides OCRs into different categories. The input to the classification can be online or offline [17][3]. The online system is a real time system that uses a handheld device that can sense the movement of hand to get the text image. A few offline systems are present and that too for Latin Language [3][17].

In case of offline or static recognition system, the already present document is scanned to take an image, or a digital image is synthetically created. This method is relatively complex as compared to offline because in online system the characters are being interpreted separately as soon as they are written while in offline whole of the text is taken as input and needs to be processed [18][3]. The text in online system is always handwritten while in offline system the text can be both handwritten as well as printed [3].

2.2.2 Writing mode:

There are two types of writing methods that a recognition system can classify. One is handwritten and the other is in printed format [3]. Both of these types have complexities associated with them. Handwritten documents pose an extreme challenge because of the variation in the writing styles of the people. There could be many possible writing styles of a single character. Printed text is relatively easier than handwritten but that too is not without difficulties. A printed text can be in many different font styles, for example English can be typed in numerous fonts like Times New Roman, Calibri, arial etc. For Arabic the writing style is Naskh while Urdu uses the Nastalique style. Among English, Arabic and Urdu the font that is most cursive and difficult to recognize is the Nastalique writing style of Urdu. Similarly, the quality of the printed document can create issues during recognition. If the image has noise, distortions or rotation it can also pose issues for the classification system [19].

2.2.3 Font

The capability of an OCR system to recognize language in only a single font is called single font optical recognition system while a system that can recognize multiple font styles is known as omni-font optical recognition system. The geometric shape of the font affects the recognition capacity of the classification algorithm [20][3].

OCR that has been trained on one font style can not recognize the same word written in another font. To get this capability, the training of the classification needs to be performed for every font style making it a time taking process. The omni font recognition systems have a generalized training process that is trained on multiple fonts [20][3].

2.2.4 Script

Script connectivity is also a differentiating factor of recognition systems. Scripts are either cursive or non-cursive. A non-cursive script has all of its characters separate i.e. they do not join to each other or do not change their shape when joined. In cursive script however, the characters of the language join together and have different shapes based on their position in the word. They have varying shapes if present in the start, middle or end of the word. Such systems are known as Intelligent recognition systems. The focus of our research is also on the recognition systems for the cursive script.

Urdu is a highly cursive language. It has 45 characters with 38 primary characters among which only 10 are non-joining characters meaning they do not join to any other character. The font is Nastalique that is written diagonally, and overlap exists between the words. All these things make Urdu a highly cursive language [7][3][17].

2.3 Research on OCR category

The OCR can be categorized based on the above mentioned 4 types. The review of the literature shows that most of the work is done on offline, printed and both isolated and cursive Urdu scripts. Though the classification accuracy is reported higher in case of separate characters but for cursive scripts the research is still going on. Following research papers have proposed methods to recognize separate Urdu character using various techniques.

In [21] an offline method has been used to recognize the separate Urdu characters. Similarly, in [22] a recognition system has been proposed that recognizes separate Urdu characters written in the Nashkh font style. [23] have also recognized single Urdu character ligatures of various font sizes. In [24] another method is used named SoftConverter that recognizes isolated Urdu characters. A pattern matching technique is used in [25] to recognize the Urdu characters written in Naskh writing style. Attempts to recognize the cursive Urdu script containing the text having words or ligatures have also been done. Following are some of the works reported to recognize the connected Urdu characters;

In [26] a dataset consisting of 56 classes having 100 images in each category of Urdu ligatures and words is used. The study works in two phases, the first phase segments the ligatures into separate characters and then these separated characters are trained using feed forward neural network.

Another work proposed by [27] recognizes Urdu ligatures in Nastalique writing style by first segmenting and then recognizing by using Finite State Models. In another work [28] 200 selected ligatures created in Noori Nastalique font by Ahmad Mirza Jamil were used for recognition. Holistic approach for cursive text recognition was used in which at first the special ligatures including dots and diacritics were recognized. Then these recognized symbols were associated to the main or primary ligature by finding out the most probable neighbor. At the end, it was fed to the feed forward neural network for final classification of the ligature. In another study [29] a segmentation based approach has been used to recognize the Urdu ligatures using Nastalique font. It also describes the difference between Nastalique and Naskh writing style and explains the reason for why the Naskh writing style could not be used for classification of Nastalique font. In [30] Recurrent Neural Networks were applied for the recognition of Nastalique script.

2.4 Research on OCR Process:

The six steps of an OCR system are listed in the diagram below

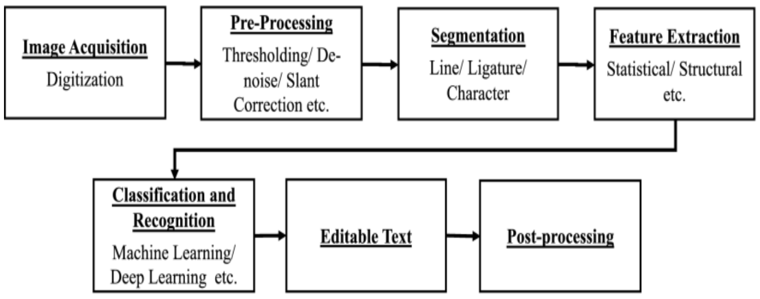


Image Acquisition
Digitization
Pre-Processing
Thresholding/ De-noise/ Slant Correction etc.
Segmentation
Line/ Ligature/ Character
Feature Extraction
Statistical/ Structural etc.
Classification and Recognition
Machine Learning/ Deep Learning etc.
Editable Text
Post-processing

Figure 2.2: Flowchart of OCR Process

In an OCR system, some, or all the above steps are involved. For example, the last step of post-processing is an optional step and may not be utilized in some OCR systems. Research has been carried out for independent steps by some authors while others have contributed to carry out research involving all the steps of an OCR process.

2.4.1 Image Acquisition:

The first step of Image acquisition that is to acquire the image. In case of offline recognition, this step always contains text which is mostly in the form of images. The images could be scanned copies of handwritten or typed documents or could be synthetically generated. The challenge related with this step includes the image quality, font size and style used for the text inside the image. A noisy image with small font and image size is likely to affect the recognition of the images during the classification phase. The small font in an image may be considered as noise, affecting the overall recognition accuracy. To get better classification accuracy, it is important that the image should be of high quality with minimum noise.

Research has been carried out on all types of text images that include synthetic, real-world images or both. [25],[26],[27], [31] and [32] used synthetically generated printed images in their research to recognize Urdu script. [25] recognizes isolated Urdu characters in Naskh font present in binary format. The images have a font size of 36. Similarly, in [26] the training set includes images of 41 printed Urdu alphabets in 56 categories that are used for recognition of compound words.

A lot of work has also been done on scanned images of printed or handwritten documents. These types of images are very difficult to deal with as the chances of noise in the image and image quality degradation during scanning are highly likely to occur. Yet, [29], [33], [34], [35], [36] and [37] have done work on scanned images.

In [29], scanned images of printed Urdu books are used. An image dataset is gathered from 500 pages of 100 Urdu books. In [33] Urdu handwritten characters and numerals were recognized using topological, contour and water reservoir concept-based features. The dataset includes scanned images of handwritten documents with some on good quality papers and some on inferior quality paper. [34] used dataset of scanned images of characters, numerals, and objects from Urdu newspapers. The images for a single character or form having three various shapes were combined into one class and used for training the network.

2.4.2 Pre-Processing:

Pre-processing is the most important step of the OCR process. In offline OCR, the images are either handwritten or printed which are either scanned or synthetically generated. In any case, the input to the OCR is an image. A noisy image can affect the classification in the recognition step of the OCR. It is very important that the image acquired should be pre-processed before feeding it to the feature extraction or classification network. During pre-processing any type of quality breakdown, distortion, clutter, or orientation issue is removed which at the end produces a good quality image providing good classification results. Some pre-processing techniques are smoothing, noise removal, thinning, thresholding, deskewing, normalization, dilation etc. [3],[37].

[36] worked on the classification of Urdu text. Urdu script in the form of separate characters and ligatures were obtained by scanning the images at 300dpi. The acquired images were single tone and they were converted to two-tone images, in the form of 0 and 1, by using a histogram based thresholding method. Presence of noise and irregular character boundaries were smoothed out to remove the noise from the images.

[38] used the pre-processing technique of smoothing, filtering and thinning on different Urdu characters. These techniques removed noise from the images and reduced the thickness and slanginess of the images. In [39] Urdu characters written in Nasakh wring style were recognized. During the pre-processing step, de-hooking and image smoothing techniques were applied to perform image binarization. initially the input image was RGB Image was converted to grey scale with pixel range between 0 to 255. Then this grey scale image was finally converted to binary image having pixel value that is either 0 or 1. To the obtained binary image processes of skeletonization and thinning were used that resulted the image pixel to be one pixel wide to be used for feature extraction.

[40] recognizes the Nastalique ligatures based on diacritics present in the script. It uses the pro-processing technique of binarization that converts the grey-scale images to binary. For line segmentation vertical histogram is used and baseline identification is performed by identifying the rows having maximum number of pixels. [41] used stretching technique for identifying isolated characters in a ligature and to remove overlapping.

2.4.3 Segmentation:

Segmentation is the process of separating the text from other components present in an image. There are three levels of text segmentation. This includes page decomposition, line segmentation and text segmentation. The first step of page decomposition involves separating out the text from other components like tables, figures, headers, footers etc. Once the text in the image is recognized the next step is to segment the lines from it. The most common method is horizontal projection profile. Others include smearing, stochastic and Hough transform. When the lines are segmented, the last and most important step is the text segmentation which could be at three levels, word, ligature (combination of one or more character) and the character level [3].

The approaches used for text segmentation are of two types, i.e., analytical and holistic approach [3]. Analytical segmentation splits the text, either handwritten or printed, into separate characters before recognition. This approach is highly error prone as it requires deep understanding of the split point. Urdu being a cursive language is highly variable and overlapping. Extensive knowledge of characters' start and end points is required but still the segmentation cannot be perfect because of complexity, size variation and position of the characters in the text [3].

Holistic approach does not perform character level segmentation rather the ligature or word is recognized as a whole. This method has shown to perform well as compared to the analytical method of segmentation. It has gained much attention from the research community and most of the work on OCRs has now shifted towards holistic ligature or word recognition [3].

Some of the work done using analytical method of text segmentation include; [26] applied text recognition in two phases i.e., segmentation and classification. The segmentation is done by using pixel strength. This pixel strength is measured to identify the different words in a line and to find out the boundary or joint of ligature/connected component.

[27] has also used segmentation before the final classification. This method first identifies the text areas within the acquired bitmap image of the document. After page decomposition, the text lines are identified and then it is narrowed down to word and finally the character level for identification.

Similarly [41] has used the analytical approach for text segmentation and the segmented text is then used for recognition. This approach applies the horizontal projection method for line segmentation of binary image. After that baseline detection is performed using horizontal projection. The next and main step of ligature segmentation is performed by using the connected components algorithm. [8] used the concept of text line identification from UTF-8 encoded images and are then classified using recurrent neural networks.

2.4.4 Feature Extraction:

After text segmentation, the most important step of feature extraction arrives. It is a very vital step as it can directly impact the classification accuracy [3]. Feature extraction involves the identification of a set of parameters called features from the dataset. The features depict the characteristics or attributes of the objects in the dataset.

There are two types of feature extraction methods which are feature learning and feature engineering. In the feature learning approach features are automatically extracted while in feature engineering, hand-crafted features are used.

Feature Learning:
Feature learning is the automatic extraction of features by the algorithms. This is extremely helpful when the numbers and type of machine learning features cannot be identified. Feature learning can be supervised or unsupervised. In supervised feature learning, prior knowledge regarding the dataset is available. Labelled train-

ing examples are present. The supervised learning algorithms use neural networks. In unsupervised machine learning, labelled dataset is not available. Some of the examples are clustering algorithms and auto-encoders [3].

Feature Engineering:
Feature engineering as mentioned earlier is creating handcrafted features from the dataset. It is a difficult process as it requires extensive knowledge about the dataset. The number, quality and quantity of features required must be determined so that the classification algorithm gives high classification performance [3]. There are three types of features in this category. This includes statistical features, structural features and global transformation and series expansion [3].
1. Statistical Features:
These features are based on the distribution of pixels in an image [42]. These features can be extracted with ease as compared to other statistical feature engineering techniques. These types of features are less affected by noise, have low complexity and can be detected at a higher speed.

Few examples of statistical features include projections, zoning, crossing and distances [42][3].
2. Structural Features:
Structural features as the name indicates are based on the structure of the character within the text or incase of objects the strokes and boundaries of the object. These kinds of features when used for extraction of Urdu language are extremely difficult due to the context sensitive nature of this language. The neighboring characters can overlap and change the shape of the character within a ligature or word [3].

Structural features are the geometric or topological properties of a character for example start point, end point, horizontal lines, vertical lines, crossing point and curves etc [3].
3. Global Transformation and Series Expansion:
These features represent the image as continuous signals. Global transformation, transform the image at first into a signal and then it is used to extract features. Series expansion is a way to represent a signal like a parallel combination of multiple smaller signals that are expanded to form one large signal. Few prominent examples of these type of features include Gabor filter and transform, Fourier Transform and Karhunen-Loeve or the KL expansion.
Research has been conducted in the field of Urdu OCR utilizing both feature learning and feature engineering approaches. In the past, most work however has been carried out using feature engineering method, but the feature learning method has gained a lot of popularity in the recent times. The dataset size and complexity has increased over time and such for such datasets feature learning has proved to give better recognition accuracies [3] [7].

In [33] separate Urdu characters are recognized by using contour, topological

and water reservoir concept based features. These features include the number and position of holes, number of different components of a feature, information regarding character's contour and number of reservoirs of different components of a contour etc. These features are then used to build a decision tree for character recognition. [36] involves the recognition of Urdu language by first doing segmentation at the page, then line and at the end at the character level. After segmentation and pre-processing, the feature extraction is done using the feature engineering approach where the following features are used;
Water reservoir principle-based features: These features utilize the concept of a water reservoir which is that if the water is poured from one side, then it flows and gets stored at the specific parts where the depth is present. These depths are considered as the water reservoirs. These features are the top, bottom reservoir, height of reservoir, water flow level and reservoir base line.

Head-line Features:
This feature depicts the words in the text line as rows of black pixel based on the head of the characters in an image. The headline with maximum number of black pixels indicate that the heads of the character lie in the same line. This feature will generate short black lines for Urdu text as the script has words with varying shapes as compared to English.

Distribution of vertical strokes features:
This is a simple feature that extracts the presence of vertical strokes in an image. For Urdu language there are few characters where the vertical strokes are present as compared to English.

Component Overlapping Feature:
This feature calculates the overlapping of one character or word into the other. Incase of Urdu script, there are many words that overlap to one another. This is a distinctive feature that is used to identify Urdu script from English.

Height Distribution of Components:
This feature identifies the height distribution of various words in a text. For Urdu text different word have varying heights due to the changing shapes of the characters based on their position in a word.

[27] uses the finite state models to recognize ligatures in Urdu text. The recognition involves the segmentation of the text at page, line and then character level. After character level segmentation three features are extracted named as the feature set. This includes the thickness, height and the angle rotation.

Similarly [39] uses the hand engineered features for recognition of separate Urdu characters. There are two phases in which the features are extracted. In the first phase, different shapes of the individual characters are identified by auto clustering using the Kohonen Self Organizing Map. In the next phase 25 features including the start point, end point, joint, width, loop, curve etc. are identified which are used for the final text classification.

[43] uses recurrent neural network and MDLSTM for recognition of Urdu language. The said method performs feature extraction to extract information from Urdu text line. A right to left sliding window of size 4 x 48 is used which slides over the text line that has been normalized to a fixed height whereas the width is variable. This sliding window extracts the geometric and statistical features which are then used for final text recognition.

2.4.5 Classification:

Classification is the process of arranging elements into different groups based on their similarity. The element to be categorized could be an image, text or object. Image Classification contains images that need to be categorized into groups known as classes based on distinct features of a class [1],[3].

In a classification algorithm, the first step is training and the second is testing. In the training phase, features from the images are extracted and fed to a classification model that learns those features. Later the trained model is used to recognize unknown data called test set into various categories [1],[3].

Image classification is used in the optical character recognition systems where textual images are used for text recognition. There are two methods used for classification. These are machine learning and deep learning [2].

Both methods are used by the research community for the classification purposes. Both have their usage as well as limitations. Among these methods, deep learning has been an active area of research for image classification in general and recognition of Urdu language in particular [2],[3].

Machine Learning:

Machine learning is the use of methods that enable the machines learn patterns like a human. It is a process that extracts meaningful information regarding the data and enables computer to understand and utilize this knowledge for identification on the unknown data. Example of machine learning algorithms are Neural Networks, Support Vector Machines (SVM), Decision Tree and Naïve Bayes method [1],[2],[3].

There are two types of machine learning i.e., supervised and unsupervised machine learning.

In the supervised machine learning approach, labelled training data is available. This data is used to create a function which can then be used on unseen data for classification. Examples are decision tree, naïve bayes and neural networks [1],[3].

In unsupervised machine learning prior knowledge is not available. Or in simple terms, there are no labelled training data examples, rather the algorithm has to identify the class labels directly on the unseen data. This method is quite challenging as the machine has to start predicting without prior information [1][3]. Example of unsupervised algorithms are nearest neighbors, hierarchical and probabilistic clustering and mixture models etc [3].

Deep Learning:

Deep learning is a type of machine learning that utilizes large datasets to extract information (features) directly from the dataset. This is quite an old method but has been utilized now extensively for classification, particularly for Urdu language recognition. It has gained much attention from the research community recently and also has proved to show better results for the classification tasks [1][3].

Difference between Machine and Deep Learning:

The fundamental difference between machine learning and deep learning is the type of features extracted from the dataset. Feature extraction is the most important task in a recognition system. Low quality features can directly impact the classification/ recognition performance of the system [1],[2],[3].

Machine Learning extracts hand-crafted features from the dataset. These features are extracted based on the domain knowledge of the data. These include statistical and structural features including pixel values, shape, orientation, size and position of the object to be recognized. It is quite a difficult task for finding out these features, as the domain knowledge should be very extensive. Deep learning on the other hand extracts features automatically discarding the issue of feature engineering, making feature extraction a rather easier task [3].

Machine learning algorithms work well for datasets that are small in size but as soon as the dataset increases and becomes more complex machine learning cannot perform the task. For large datasets deep learning is the key. The size of the dataset also requires corresponding computational power. For small datasets in machine learning simple computational machines can perform well but for deep learning Graphical Processing Units or GPUs are the essential requirement. Otherwise, the recognition task cannot be performed [1],[3].

Deep learning is an end-to-end process which performs object detection and recognition as a single process and ultimately takes more time to train. On the other hand, machine learning is fast at training but may take more time to test.

The major criteria that make a clear difference between machine learning and deep learning and is also a reason for deep leaning to be the latest hype, is its performance. Deep learning algorithms have proved to perform brilliantly well for the recognition tasks, providing near human perfection. The reason could be the automated feature extraction method that is utilized by the deep learning algorithms which reduces the error rates that may have been occurred by hand crafting the features in machine learning [3].

2.4.6 Contributions for Urdu text recognition using Machine and Deep Learning:

The datasets are increasing in size day by day. The amount and rate at which data has been adding up has shifted the use from machine learning to deep learning. Such large datasets cannot be handled by machine learning algorithms and also the excellent performance efficiency of the deep learning algorithms has made it an active area of research for the research community for the recognition tasks [3],[7].
Following is some of the research carried out using machine learning algorithms for recognition of Urdu text; In [21] isolated Urdu characters are utilized by extracting features and then using these features to train a feed forward neural network model. This system has achieved 98.3% average accuracy at the character level.

[24] a prototype known as SoftConverter is developed, that uses simple database to recognize the Urdu characters. The input to the system is images in the form of matrix. These images are then preprocessed by de-skewing and binarization. The binarized images are segmented at line level using horizontal histogram technique. Features including height, width and checksum (the sum of black pixels forming the body of the character) are extracted and then based on these features a database is searched giving out the most similar image. Giving out 97.43% accuracy in identification of isolated Urdu characters.

In [38] a two step classification is used for the recognition of Urdu characters. 36 Urdu characters are recognized into seven classes by using fuzzy features and by combining the recognition methods of Fourier transform and neural network. The interest regions are identified using Fourier transform and after that a back propagation neural network is used to recognize individual characters. [23] is a machine learning based font size independent method to recognize the Noori Nastalique script of Urdu language. The method first resizes the images present in various sizes to uniform size by using Splines. The resized image is scaled to make the font size same on which the OCR is trained. From this scaled image the character boundary is extracted and then filled with black pixels to be recognized.

[44] used decision trees for character recognition. A database of 441 Urdu characters of both handwritten and printed documents in scanned form are present. The system performs preprocessing techniques of noise removal, binarization, normalization, skew correction and edge detection. From the preprocessed image features of Hu moments, Zernike moments and PCA are extracted. The feature vector is passed to the J-48 Decision Tree algorithm that recognizes the input image. Among all the features used Hu moments gave the best classification accuracy of 92.06%.

[45] uses principal component analysis technique for Urdu character recog-

nition. Two databases are setup named as 'TRAINDATABASE' and 'TEST-DATABASE'. Eigen values and Eigen vectors are calculated for the images and only the maximum eigen values and vectors are kept. Feature vector for images in the TRAINNDATABASE are generated. Same is done with test set and the vector that has the minimum distance to the train set is recognized as the character. A recognition rate of 92.6% is achieved using this method.

Recognition of Urdu text using deep learning has been carried out actively. Following are few of the studies done to recognize Urdu characters, words or ligatures using the deep learning approach:

[26] uses a method to recognize the compound Urdu words. This method carries out segmentation and then classification. In the segmentation phase, the pixel strength of the text image is measured. This information is used to detect the word boundaries and joints of the various characters. The segmented image is fed to feed forward neural network for recognition. The network is trained on dataset of Urdu words and ligatures having 56 classes with 100 image samples in each class. This method recognizes Urdu characters and ligatures with a recognition rate of 70%.

A deep learning approach has also been used by [8]. In this paper, the cursive Urdu script is recognized along with non-cursive Latin script. Focusing only on Urdu text recognition, Dataset for Urdu text recognition used was, Urdu Jang Dataset. A variant of RNN called BLSTM (Bi-directional Long Short Term Neural Networks) is used with CTC output layer. The activation values from the BLSTM are utilized for alignment to the target values. In [30] offline printed Nastalique Urdu text was recognized. This method applies preprocessing to the input images which are fed to RNN having architecture of BLSTM with CTC output layer.

The trained network was evaluated on synthetically generated UPTI dataset with two variations. In the first, the character's shape variation was ignored and in the second, considering the shape variations. The error rate was 5.15% and 13.6% respectively.

[46] employs the RNN having MD BLSTM with CTC output and ANIFS method. A dataset for handwritten Urdu text images which is a publicly available RNN library is used to train MD BLSTM network. ANIFS is the forward pass and backward pass. The trained model is evaluated on UPTI dataset with an error rate of 5.4%.

The technique for recognition of offline cursive Urdu text used in [47] employs the Multi-dimensional (MD) Long-Short-Term (LSTM) Recurrent Neural Networks (RNNs) with Connectionist Temporal Classification (CTC) output layer. The method is evaluated on UPTI dataset using various experimental settings with the maximum classification rate of 96.4%.

[48] employs a hybrid deep learning method for classification of cursive Urdu script. Implicit feature extraction is a better approach to feature extraction than

the hand-crafted features. Based on this principle, a hybrid method for feature extraction is used in this study where the low-level features are extracted by using the Convolutional Neural Networks (CNN), these low-level features are fed to Multi-Dimensional Long Short Term Neural Network for extraction of high-level features. This method was evaluated on UPTI dataset for 44 different classes giving a classification accuracy of 98.12%.

2.5 Analysis:

Urdu OCR research has gained attention in the last two decades. There is a lot of room for progress in this field of pattern recognition and document analysis. This chapter has covered all the aspects of Urdu OCR. The Urdu dataset developed so far have been stated in detail. The types of OCR system, the process with which an OCR system recognizes a character has been explained in detail. Each of the detail of the type and working has been detailed along with the work done in the past in that process has also been stated.

The literature review of Urdu OCR clearly shows that the focus of the previous research was on offline, printed and isolated Urdu character recognition. The method of recognition was also focusing more on the machine learning approach that used feature engineering methods like statistical features, structural features and Global Transformation and series expansion methods. Deep learning is a recent trend in case of Urdu OCR for the recognition of compound Urdu characters including words and ligatures. Isolated characters are 45 in numbers including the dots and diacritics marks. Single character recognition has matured and given high recognition accuracies. But this is not a very remarkable achievement. Urdu language is cursive meaning the shape of one single character varies based on the position in a document. When single character recognition is applied to text lines or ligatures, the recognition accuracies are very low because the text line needs to be segmented first for recognition, and segmentation as understood by reviewing the literature, is not a suitable technique because it is error prone. The error prone segmentation technique is not a good choice for Urdu OCR systems. Considering this and the ligature recognition in mind, the next focus of research for Urdu language is the development of OCR for connected Urdu words using deep learning.

The OCRs for Urdu language now in the highlight are offline, printed, holistically segmented which use deep learning for recognition. Analytical segmentation with machine learning is not utilized as a recognition method recently.

Another research gap is the presence of benchmark Urdu Dataset. Literature review shows that efforts have been made for the development of benchmark datasets by development of datasets like CLE, UNHD, UPTI and EMILLIE, to name a few. But still these datasets can not be considered as standard to represent the Urdu language. Research needs to be carried out for development of benchmark dataset along with the recognition of dataset using Deep Learning for producing better recognition systems.

The focus of our research is development of Urdu language dataset and its recognition. The next chapter explains the method used to solve the said problem by using a deep learning method of Urdu recognition.

Chapter 3

Methodology

This chapter explains the methods used to solve our research problem of Urdu ligature recognition. Starting with the background information of Urdu language OCR and moving to the research gap present, this chapter then gives a thorough explanation of the data collection methods along with the tools and techniques used to solve the research problem.

3.1 Background

Optical Character Recognition is a process that categorizes the input text images into their pre-defined categories [1][3]. Recognition is a very easy process for humans. They can interpret an object very easily and can define and categorize it very quickly. But for machines this is a very challenging task [1]. Many high-level computational methods need to be used for the correct recognition. The complexity of this problem makes it a prominent research area. The OCR for languages such as English and Arabic have a long history. The first OCR for English Language was developed in the early 1950s [6]. Later many OCRs have been developed for various other languages.

Research on the Urdu OCR started in the year 2000 [7]. The gap in the research work on Urdu is because of the lack of Urdu dictionaries, tools, experimental fundings and the lack of benchmark dataset [3]. This lack of attention to Urdu recognition resulted in absence of any remarkable achievements for OCR for Urdu language in the field of computer vision and machine learning [2].

3.1.1 Recognizable Units in Urdu OCR

The origin of Urdu language and the complexity of the Nastalique script has been discussed in detail in the introduction chapter of the thesis. Focusing mainly on the recognition process for Urdu script, the recognizable units for Urdu language will be discussed in this section. The challenges in the recognition of these units will also be discussed.

Nastalique script is very complex, there are many challenges faced while recognizing this script. Its context sensitive nature, overlapping, diagonality and cursivity makes it a very difficult recognition task [37]. Its research started just two decades back and that too with only better recognition system for isolated Urdu characters.

The choice of the units that could be recognized by an Urdu OCR include:
• Character
• Ligature
• Word

Character Recognition:
The literature review suggests that most of the work done in the past was on the recognition of isolated Urdu characters. Recognition of isolated characters though has given high accuracy but has very less practical significance. There needs to be a higher recognition unit so that it adds a significant addition to the OCR systems [6][37].

Recognition of isolated characters requires the input text line or literature to be segmented and there are a lot of problems that can arise due to the segmentation of the words into characters for recognition. Few of the challenges related to text segmentation are listed below:
• Segmentation of words into separate characters is very difficult in the case of Urdu language. There are no fixed points that need to be cut to obtain separate characters from words or ligatures. In other languages, for example English, the characters are separated by white space that act as the separating criteria for the alphabets. Similarly in Arabic language the segmentation point can be easily detected from the headline. But in Nastalique script the characters have irregular space between them and so it is not easy to determine the segmentation point [6], as seen in the figure;

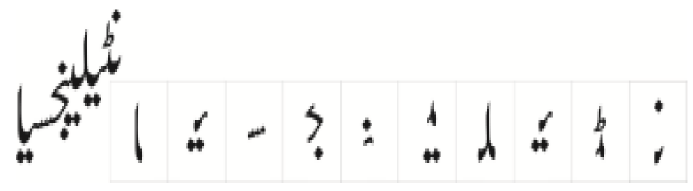


Figure 3.1: Segmentation of Ligature

• Size variation of the non-joiner characters in their various forms make it an error prone task. The character when comes isolated or at the end of a character has varying shape whereas when it comes in the start and at the end it is of small size and that too can be affected by the neighboring characters [6].
• The segmentation needs to be performed both horizontally and vertically in some cases [6]. as can be seen in the following figure;
• Size variation of the non-joiner characters in their various forms make it an error prone task. The character when comes isolated or at the end of a character has varying shape whereas when it comes in the start and at the end it is of small size and that too can be affected by the neighboring characters [6].
• The segmentation needs to be performed both horizontally and vertically in some cases [6]. as can be seen in the following figure;
Due to the issues that occur because of the segmentation required for character

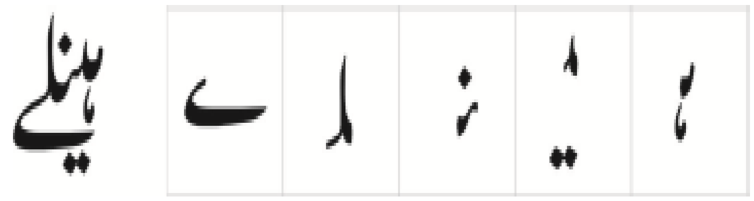


Figure 3.2: Horizontal and Vertical segmentation of Ligature

recognition, almost all the research has been shifted towards the holistic method of text segmentation, where the recognition unit is ligature. Ligature is a combination of one or more characters that may not form a complete word. The ligature does not need to be segmented and is recognized as it is.

Though ligature recognition has eased out the problem of text segmentation but the are many other challenges associated with Urdu ligature recognition. Few prominent issues are; The total number of ligatures that makeup the Urdu language is still unknown [6]. This is due to the lack of any benchmark Urdu dataset. Few attempts have been made to create Urdu datasets like CLE, UNHD and UPTI, to name a few. But the requirement of a benchmark dataset still exists [3].

3.1.2 Recognition methods for Urdu Classification:

The review of literature suggests that most of the previously recognized Urdu datasets contained isolated Urdu characters and the method used to recognize those characters were based on machine learning approach. The machine learning algorithms like Decision Trees, Naïve Bayes, Neural Networks were used for the character recognition [3] [7].

The choice of recognition unit focused now for Urdu language are ligatures. The exact number of ligatures that cover the Urdu language is still unknown, but the datasets used so far indicate that here are thousands of ligatures that cover whole of the Urdu corpus. For such large datasets, machine learning algorithm cannot perform well. For recognition of large datasets, deep learning models are used [3].

The focus of the research in the field of Urdu OCR is now towards holistic approach of text recognition using deep learning methods.

3.2 Research Question

Urdu script recognition, as stated before is a recent focus of the research. There are no sophisticated OCRs available for Urdu language [3][6][7]. Recognition of holistic Urdu script using deep learning is an under-research topic. The research carried out so far has not produced very promising results.

The first problem is the absence of benchmark dataset [6]. The recognition methods applied, used datasets that were privately created and the results on them cannot be considered as standard.

Secondly, the recognition rates using existing methods are not very high. There is a need to do further research on this topic. For this purpose, we have created an Urdu ligature dataset that covers all the possible ligatures of the Urdu language. Secondly, we have applied deep learning method over this dataset for holistic recognition. We are hopeful that the results of this recognition process will be a step forward towards creating an efficient recognition system for Urdu and the generated dataset may contribute towards building a benchmark dataset.

3.3 Dataset Generation and Related Issues

The following section of this chapter solely explains the proposed methods and techniques used in the recognition process for the Urdu ligatures. Every detail, starting from data generation to its preprocessing and the final recognition are stated and explained in detail.

3.3.1 Urdu Dataset Generation

The existing Urdu datasets, which include CLE, UNHD, UPTI, UPTI 2.0 are few prominent datasets. These datasets though large enough do not qualify to be the standard for Urdu recognition [3][13][15].

These datasets include words from various domains including sports, entertainment, news, finance and consumer information etc [16]. But these datasets do not cover all the possible ligatures and hence the recognition systems built on these datasets are not 100% accurate.

Owing to this absence of benchmark dataset for Urdu. A dataset has been created under the supervision of Dr. Khawar Khurshid. The dataset has been named as ‘CEFAR’ Dataset. This dataset contains exhaustive combinations of Urdu characters covering all the possible ligatures.

Characteristics of Dataset:

• Binary images formed by exhaustive combination of Urdu characters. Exhaustive means all the possible combinations of Urdu characters. Initially the combinations of characters forming ligatures have length of 2 and 3. Meaning, the possible combinations that could occur in length of 2 and 3 from all the Urdu characters.

For example, in 2-character exhaustive combination, image is a ligature having 2 characters. Like, the 2-character ligature combination of Urdu character ‘Bay’ to all the other characters. This forms the 2-character ligature set for character ‘Bay’ and similarly for all the other Urdu characters in the Urdu language.

• This dataset contained binary images, each having a size of 600×500 pixels. Each image has a size on the disk of 4 KB and the total dataset has size of almost 2 GB.

• Initially the total ligature images for 2 and 3 characters were 32480.

• Various augmentation techniques were applied on the train data to increase the number of images for classification using deep learning. These augmentation techniques were dilation, erosion, rotation, S&P noise and transformation. This led to an increase in the size of the training data. Each ligature had 14 different forms after applying augmentation.

• The dataset is divided into 2 and 3 characters’ ligature sets separately, as well as combined. Detail of the number of images in each ligature set is as follows.

Ligature Sets	Number of Ligatures (Binary Images)
2- Character Ligature Set	15680
3- Character Ligature Set	439040
Combined 2- and 3-Character Ligature (Final Dataset)	454720

Table 3.1: Number ligatures in each ligature set.

3.3.2 Class Imbalance Issue

Deep learning algorithms have shown to perform brilliantly well for the classification and recognition tasks. In the past studies CNNs have provided state of the art results in case of multiple classification problems. Though these networks give high performance efficiency yet one of the major requirements of these deep learning networks is to have large datasets for training. The deep architecture of these networks having millions of neurons, requires large enough datasets with uniform class balance to perform well. This challenge of having large well-balanced dataset is our goal [10].

Our dataset that we generated for the recognition task also faced from the class imbalance issue.

Each of the ligature formed a separate class because none of the ligatures was same or had similar shapes. In Urdu ligature the dots and diacritic marks, if considered make every ligature to have a different structure and shape. Therefore, none of the ligatures could be put together into a single class. This caused our dataset to have 32480 classes and same number of samples in each class. So, the issue of small dataset having low intra-class variability and high number of classes emerged.

High Number of Classes and Low Intra-class variability:
Today's world is in era of 'Big Data'. The size of the datasets has increased to such an extent that the only solution to handle such a large data lies in deep learning. One of the prominent examples of large dataset is ImageNet Dataset that contains 10,000,000 training images divided into around 10,000 classes [49].

The high number of images is not an issue in case of deep learning algorithms rather it increases the model accuracy by providing large set of features. This large number of training data becomes a challenge when the number of classes also increases. This leads to data having large number of classes but small number of features. When the number of samples in a class are small or imbalanced it provides small set of features, the algorithm cannot train well, class bias increases and causes the model to over fit [49].

Same was the issue with our dataset. The dataset had large number of classes with imbalance data. This led to class imbalance. If such a data would have been used to train the classification algorithm, it would have caused low classification accuracy of the deep learning model.

3.3.3 Data Augmentation

One of the best solutions to solve the problem of class imbalance is to perform data augmentation. Data augmentation involves the use of various transformation techniques over the data to synthesize new images. The transformations include rotation, zoom in and out, creating reflection of the original image, distortions or change in the color pallet [10].

We also applied various techniques of data augmentation to increase the number of images in each class. The techniques applied over the original image to create new images were, dilation, erosion, rotation, S&P noise and transformation.

This method increased the number of images in each class but still the number of images was not enough to avoid the overfitting issue during the recognition process. This is because the number of images was not large enough that could completely represent a class. And the number of classes was still very high. The number of features needed to be increased to achieve high generalization of the deep learning network.

3.3.4 Redefinition of classes

The requirement to increase the number of representative images in each class and to reduce the number of classes, we used another method to redefine the classes based on the structural similarity of the ligatures.

The method to combine characters having the similar shapes has been reported in literature. For example, in [48] characters having similar shape variations i.e., isolated, start, middle and end character is given single class.

A ligature is composed of two components. These are;

- Primary component: It is the basic shape of the ligature
- Secondary component: These include the dots and diacritic marks.

The dots and diacritic marks when used over the primary component, separates out a specific character or ligature from the others. If the dots and diacritic marks are removed from the character, then many of the characters will have the similar shape because they share the same primary component. We used the same approach to redefine our classes. The ligatures having the similar shapes were grouped together in a single class while their secondary components were ignored.

For example, the 14 ligature images of each class i.e. ‘ain-ain’, ‘ain-ghain’, ‘ghain-ain’ and ‘ghain-ghain’ were grouped together. The structural analysis of the ligatures within these classes show that all these classes share the same primary ligature. By combining all these images from 4 different classes into a single class increased the number of images in this particular class to 56 and also reduced the number of classes.

This method greatly reduced the number of classes. Initially the number of classes in the training data were 31360. The newly generated classes were 3116.

This reduced the number of classes to approximately 90%.

3.4 Dataset Generation Process

In the previous section, an overview of the Dataset Generation was given. The above mentioned methods including the initial generation of ligatures, data augmentation and the redefinition of classes is explained in detail in the sections below. The complete steps in each of the methods used are elaborated.

3.4.1 Character Labelling:

isolated character was first given a number, based on the numbering scheme. As we know that Urdu characters are joiners and non-joiners, So in the 2 character ligature, there were only 2 characters that join together i.e. initial and last character. While the 3 length ligature contained characters at initial, middle and last positions.

3.4.2 Ligature Labelling:

The initial images for the ligature combination of 2 and 3 character were generated using MATLAB.

A ligature as stated before, is a combination of one or more characters. In our case we chose the ligatures which have length of 2 and 3. At first all the ligatures having length 2 were generated. The name of each ligature was given based on the number of the character being joined together.

For example,the combination of character 'bay' with 'alif' having image name 2_1_1_0_0_0 was given the name bay-alif.

Image name:
Images when generated using MATLAB were given a numeric label. This label had the generic form;

'Ligature Length_ Initial Character_Middle Character_Last Character_0_0_0_.png'

For example , Image 3.3 when generated, was given the numeric name 2_1_4_0_0_0_*.png* where, 2 refers to the ligature length, 1 is the number of the initial character and 4 is the number of the final character. Rest of the numbers are given 0 because no character is present at that position of the ligature.

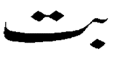


Figure 3.3: Example image showing naming scheme of the ligature for 2 character length ligature

Similarly, 3_17_3_8_0_0_*.png* is the numeric label for the 3 character length ligature. Here 3 shows that the ligature has 3 characters in it. 17 is the number of initial character, 3 represents the numbering of the character in the middle position and 8 is the number of the character at the final position.

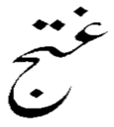


Figure 3.4: Example image showing naming scheme of the ligature for 3 character length ligature

3.4.3 Data Augmentation:

The initial number of images generated by simply taking the exhaustive combinations of 2 and 3 character length ligatures were;

2-Character Ligature Set	1120
3-Character Ligature Set	31360

Table 3.2: Initial number of images in the 2 and 3 character ligature set

These images were not enough to be used in a deep learning algorithm for training. To increase the number of images, data augmentation technique was used. In this method the following image transformations were applied over each ligature image. A single ligature image increased in number to 14 images each.

The image transformations were;
• Dilation
• Erosion
• Transformation
• Rotation
• S&P Noise
This increased the number of images as following;

Ligature Sets	Training Data
2-Character Ligature Set	15680
3-Character Ligature Set	439040
Combined 2 and 3 Character Ligature Set (Final Dataset)	454720

Table 3.3: Number of images in the 2,3 character ligature set (seperately and combined) after Data Augmentation

Due to the application of data augmentation techniques, the new name of the images became;

'Ligature Length_ Initial Character_Middle Character_Last Character

So, the new names for the images in the Figure 3.4 and 3.5 became 2_1_4_0_0_0_$dilation_1.png$ and 3_17_3_8_0_0_$dilation_2.png$.

3.4.4 Limitation of the dataset:

The image dataset finally created had 454720 ligatures in the form of images. The number of images was good enough to train a machine learning algorithm for classification. But there were limitations associated with this data set, which was Low Intra class variance and High number of classes

Low Intra class variability refers to less number of representative samples in a class. Therefore, the training model has very few features to learn. When such a dataset is used to train a deep learning algorithm, it causes the model to overfit. As there were few training examples in a class, the generalization of the training model decreases [51].

Moreover, the dataset had 31360 classes, such a high number of classes with only 14 image samples in each class would give low classification performance of the network. As this model would be highly biased and will not be able to predict well on unseen data.

If this dataset would have been used to train the network, the result would be a model with higher training accuracy and low test data accuracy. Such an overfitted and biased model would not have produced significant results for the recognition task.

3.4.5 Solution to the Limitation:

The solution to this low intra class variability and high number of classes was either to use data augmentation [10] or to reduce the number of classes. We had already used the data augmentation method for increasing the training data. So, this method could not be used again, as the number of images in the training data were good enough for training a deep learning algorithm. We used the second approach, which was to reduce the number of classes by combining multiple classes. This method helped to reduce the classes as well as increased the number of samples in each class.

To do this we used the approach of combining the ligatures based on their primary component. A close analysis of the structure of the ligature shows that even though ligatures have different secondary components, they do have the same primary component.

The process to reduce the number of classes, if done manually was a very tedious task. To reduce the time complexity and to automate the process, a python program was written on Jupyter Notebook to gather all the images with similar ligature into a single class. Though, the numeric labels of images could also be used as class labels but a more realistic labelling scheme, giving Urdu names, was used for giving labels to the classes.

The technique used to merge the similar primary component ligatures, was to first re-label the initial, middle and final position characters. After character re-labelling, it became lot more easier to combine the ligatures. When the similar primary structure characters were given the same name, the ligatures formed from them automatically got the same label. The images with same ligature label were grouped into a single class.

After grouping the characters based on the same primary component, the label of every character within the same class was given the label of the first character in that specific class.

By re-labelling the characters after grouping based on similar primary component, we directly reduced the number of classes of ligature. The process was done by using a pyhon code. The algorithm of the code was as follows;
• Initial labels of the ligatures were extracted.
• The character numbers were seperated for each image. The character numbers were given the new name as per the rules defined in the tables 4.9 and 4.10.
• After forming new name for the ligatures the image was copied to a new folder

with the new class name.
• This process was repeated for each image, giving the classes being grouped according to their structural similarity.

Renaming based on similar structure at the character level automatically caused re-labeling at the ligature level. Without manually identifying similar ligatures of length 2 and 3, we simply gave the characters, within the ligatures, similar name based on their structural similarity and when the characters were renamed, the images of the ligatures that had the same name were grouped together into a single class. This way the number of ligature classes automatically got reduced without inspecting each ligature separately. The final number of classes formed by the class reduction method based on primary component similarity were 3116. This reduced the number of classes from 31360 to 3116 only.

3.5 Pre-Processing

The step prior to training of the deep learning network is the preprocessing of the image dataset. Pre-processing involves use of various operations such as noise removal, thresholding, smoothing and de-skewing etc. to create good quality images that improve the recognition accuracy of the classification model and reduce the training time [3].

3.5.1 Image Details

Each image in our Urdu dataset had the following properties;
Image Type: PNG
Image Width: 600
Image Height: 500
Bit Depth: 1
Color Type: Gray Scale
Image Size: 600 x 500

3.5.2 Pre-processing Method

The images in our dataset were already binary images and therefore there was no need to convert them from RGB to grey scale. The images were already normalized to pixel values of 0 and 1. Random resizing was applied to resize the images into size of 224 x 224. The original size of the images was 600 x 500 which was quite large and could not be used directly for training as it leads to computational complexity during the training phase of the recognition.

Random augmentation function was also applied over the images to induce augmentations over the images before training the deep learning network.

3.6 Training

Deep learning models make use of deep neural network architectures for classification. Use of deep learning for image classification is the latest and an active area of research [3]. A lot of work has been reported in the literature which makes the use of deep learning models for image classification providing excellent classification accuracies.

We also made the use of deep learning for ligature recognition. The proposed method to recognize the ligatures made use of transfer learning approach utilizing the Resnet34 architecture.

3.6.1 Transfer Learning

It is the approach of network training that makes use of already trained model on a dataset in related domain for predicting outcomes on dataset present in another distribution and domain. This method saves from the time complexity for training the network architecture from scratch. Rather, the pretrained models are used for recognition on a dataset in a similar domain. This way it saves the time for training the model on custom dataset from scratch. It is a highly used method in deep learning and NLP tasks that gives high classification accuracy with less training time [52].

Resnet34 is a deep learning model that is pretrained on ImageNet dataset that contains 1.3 million images. The pretrained model is loaded and then weights are updated using our own dataset, to train the model quickly, reducing the training time [52].

3.6.2 Need for Resnet

Resnet models are used in the recent deep learning techniques to overcome the gradient vanishing problem. Deep neural networks have large number of layers that learn the features starting off with low-level features at the initial layers and moving on to higher layers that learn most of the high-level features. But there is a threshold to the length of a deep learning network. If the number of layers become very deep, the performance of the model decreases. The reason behind the performance degradation is that in a very deep architecture, the gradient function for loss calculation gradually goes to zero. This causes the weights of the network to stop updating. This issue is resolved using the residual neural networks it can directly skip some of the layers and can directly move to the next layers solving the problem of vanishing gradient [53].

3.6.3 Resnet34 Architecture

Resnet34 uses the basic 34-layer plain network architecture. It is inspired by the VGG-19 model to which shortcut connection is added. These connections convert the plain network to residual network [53]. The first convolutional layer has 64 filters with kernel size of 7x7 followed by a max-pooling layer. The stride size is 2 in both the cases. In the next convolutions, there are convolution layers that are grouped in pairs. The residuals are connected every 2 layers as can be seen through the arrowheads. The 3 pairs of convolutions are of size 3x3 having 64 filters, after this the next convolutional pairs are again of size 3x3 but this time with 128 filters and the pairs are repeated 4 times. After this, the next 6 convolutional pairs are of size 3x3 with 256 filters with the first layer having a stride size of 2. Followed again by three pairs with same 3x3 size but with filter size of 512. These layers keep on repeating until the average pooling layer followed by the fully connected layer.

A better explanation of the Resnet34 architecture is shown in the following figure; This figure shows that the Resnet34 model has total of 5 blocks. This first block has two layers, the first is convolutional layer of size 7x7 followed by a 3x3 max pool layer both having stride 2. The conv_2x has 3 pairs of convolution layers of size 3x3 with 64 filters, followed by 128,256 and 512 filters in the next blocks of convolutional pairs which are repeated 4,6 and 3 times in the conv3_x, conv4_x and conv5_x respectively.

3.6.4 Libraries and Dependencies

• Pytorch
• Fastai
• Jupyter Notebook

Pytorch: Pytorch is an open source machine learning library built on top of torch framework for carrying out various deep learning and natural language processing tasks. It provides various tools to support machine learning codes, handle debugging and provide GPU support [54].

Fastai: Fastai is a deep learning library that provide high level components to carry out the deep learning tasks. It provides many high and low-level APIs along

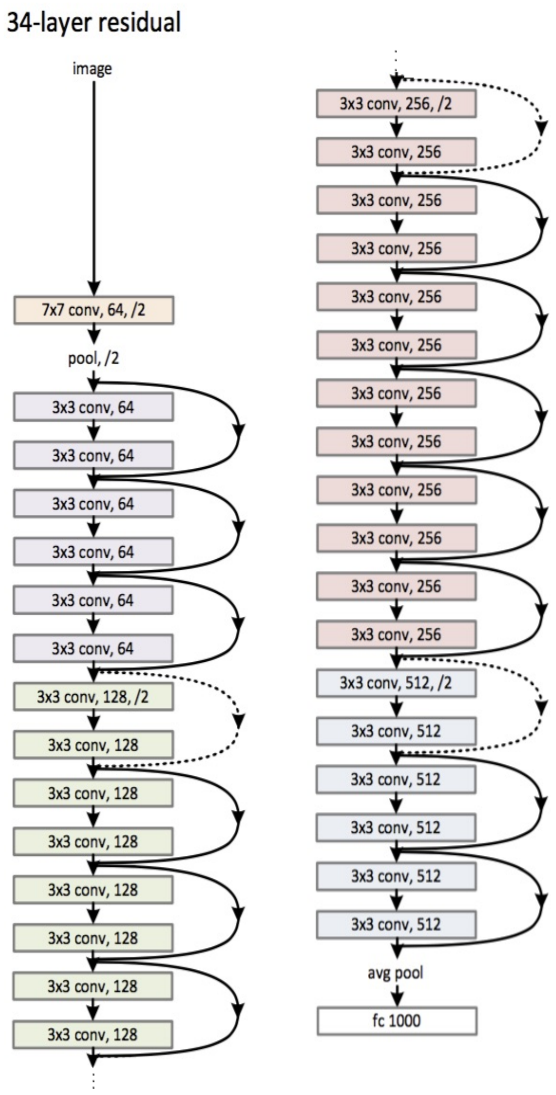
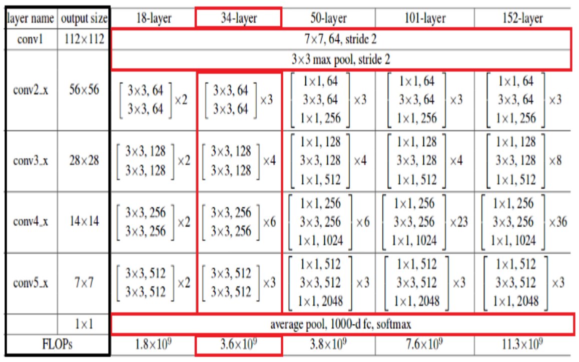


Figure 3.5: Resnet34 Architecture

with providing GPU support to carry out complex deep learning operations with flexibility, efficiency and speed [55].

Jupyter Notebook: It is a widely used interactive web based applications for writing, debugging machine learning codes.



layer name	output size	18-layer	34-layer	50-layer	101-layer	152-layer
conv1	112×112	7×7, 64, stride 2				
		3×3 max pool, stride 2				
conv2_x	56×56	$\begin{bmatrix} 3\times3, 64 \\ 3\times3, 64 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 64 \\ 3\times3, 64 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times3$
conv3_x	28×28	$\begin{bmatrix} 3\times3, 128 \\ 3\times3, 128 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 128 \\ 3\times3, 128 \end{bmatrix}\times4$	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix}\times4$	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix}\times4$	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix}\times8$
conv4_x	14×14	$\begin{bmatrix} 3\times3, 256 \\ 3\times3, 256 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 256 \\ 3\times3, 256 \end{bmatrix}\times6$	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix}\times6$	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix}\times23$	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix}\times36$
conv5_x	7×7	$\begin{bmatrix} 3\times3, 512 \\ 3\times3, 512 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 512 \\ 3\times3, 512 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix}\times3$
	1×1	average pool, 1000-d fc, softmax				
FLOPs		1.8×10^9	3.6×10^9	3.8×10^9	7.6×10^9	11.3×10^9

Figure 3.6: Resnet34 Architecture in block form

3.6.5 GPU Requirement

Deep learning requires GPU (Graphical Processing Units) support in order to carry out the computationally complex machine and deep learning tasks. GPU supercomputers allow faster processing because of parallel GPUs that can perform complex computations. The highly parallel nature of these GPUs makes the computations faster and more efficient [3].

The classification of our Urdu ligature dataset also required the GPU support for network training and testing because of the large size of the dataset that cannot run on local machines. NVidia Tesla T4 GPU server was utilized for the GPU support. This facility of GPU was granted to us generously by the NUST Interdisciplinary Cluster for Higher Education (NICHE). We are thankful to NICHE for providing us with this facility that helped us carry out our research.

3.6.6 Resnet34 Training

Our Urdu ligature dataset was present in three forms. 2-character ligature set, 3-character ligature set and combined 2 and 3 ligature datasets. Resnet34 model was

Datasets	Training Data (Binary Images)	Validation Data (Binary Images)
2- Character Ligature Set	12544	3136
3- Character Ligature Set	351232	87808
(Final Dataset)	363776	90944

Table 3.4: Number of Images in 2 and 3 character ligature sets (Separate and Combined)

trained on all the three sets to check whether any difference on the classification accuracies occurs or not. The dataset was split into 80:20 ratio, where 80% of the data was taken for training and 20% for validation. The data was taken in batches and total of 10 epochs were applied on the datasets for training the deep learning model.

The trained models were saved to predict on the unseen data called the test set.

3.6.7 Test Data Generation

After successful training and getting good validation accuracy on the validation data. The next step was to check the generalization capability of the trained model. A higher test data accuracy shows that the model is well trained to identify the unknown dataset. A model with high validation data accuracy and low test data accuracy is overfitted and biased. Such a model cannot be considered as a high-performance deep learning model [3][11].

In our research, three test datasets were created. The procedure of creating the test data for all the three cases i.e., 2 character ligature set, 3 character ligatures set and 2 3 character ligature set combine (Final set) was the same. The procedure was;

• Splitting the data randomly into 20% test set from the whole dataset.
• Applying random augmentation over the split 20% of the data. The augmentations were not used to increase the number of images rather the new images generated after augmentation were used to replace the original images obtained during split done in the step 1. The augmentations applied were;
1. Rotation
2. Zoom
3. Width shift
4. Height shift
5. Shear
6. Noise
7. Brightness

The augmentation technique was chosen randomly for each image and after augmentation the original image was updated.
• The newly created images were subjected to resizing and random cropping the same way the train data was pre-processed. The test data images were converted to a size of (224,224).
• The test dataset finally created after augmentation and resizing was used for predicting over the trained Resnet model.
• The results of the predictions were compared and a final test data accuracy was computed.

The results including the validation and test data accuracies along with the evaluation being performed on the obtained results is discussed in detail in the next chapter.

Chapter 4

Results

This chapter explains in detail the findings and results of the methods used for the classification of Urdu ligature dataset. The method and techniques that were utilized during all the phases of Urdu image data classification including data generation, data augmentation, class re-definition, pre-processing, training and testing has been explained in detail in the methodology chapter. The focus of this chapter is on the outcomes at each of the phase of the image recognition task. The research question was the classification of the Urdu ligature dataset by using deep learning, catering the class imbalance issue with data having low intra-class similarity and high number of classes. This chapter entails the findings and the performance of the Urdu ligature recognition task for dataset with exhaustive ligatures of length 2 and 3.

4.1 Urdu Ligature Dataset

The dataset that we generated had exhaustive combinations of all the Urdu characters creating all the possible Urdu ligatures of length 2 and 3. The images were generated using MATLAB. The initial dataset contained the following number of ligature images:

Ligatures	**Number of Ligature Images**
2 Character Ligatures	1120
3 Character Ligatures	31360
Final Dataset	32480

Table 4.1: Initial Number of images in each ligature set

4.2 Dataset Generation Process

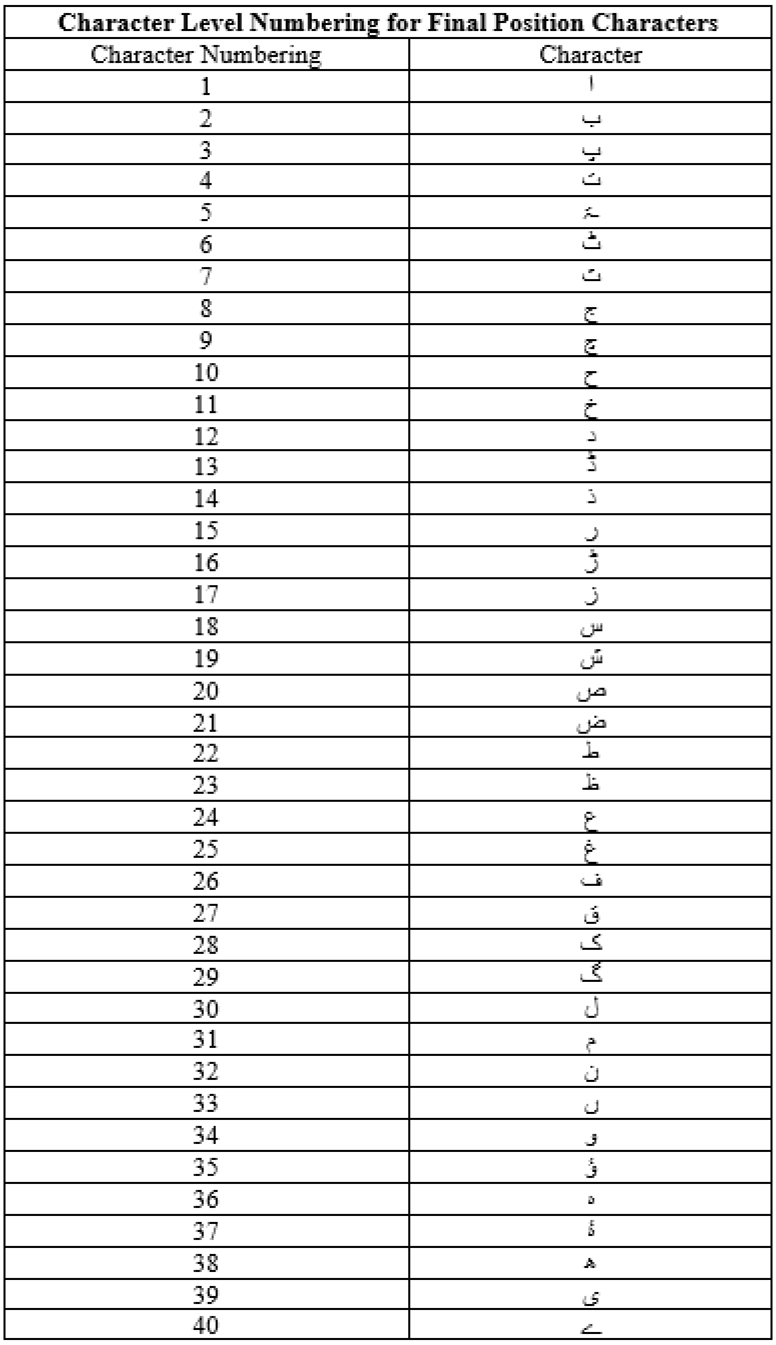
4.2.1 Character Level Labelling

The generation of Urdu ligatures was not a single step process. Rather it started from the lowest level i.e., characters. The characters were numerically labelled first. They were then joined to form a ligature based on the given numeric labels. The reason behind giving numeric labels to characters was to ease out the ligature generation process.

As already stated, ligature is a connected component of one or more characters. Therefore, all the characters within 2 or more length ligatures are all joiner characters. So, the Urdu characters exist in a ligature at three positions only, these are initial, middle and final. The character labelling at initial and middle position are shown in table 4.2 and the final position labels are shown in table 4.3.

Character Level Numbering for Initial and Middle Position Characters			
Character Numbering	Character	Initial Position Shape	Middle Position shape
1	ب	ﺑ	ﺒ
2	پ	ﭘ	ﭙ
3	ت	ﺗ	ﺘ
4	ٹ	ﭨ	ﭩ
5	ث	ﺛ	ﺜ
6	ج	ﺟ	ﺠ
7	چ	ﭼ	ﭽ
8	ح	ﺣ	ﺤ
9	خ	ﺧ	ﺨ
10	س	ﺳ	ﺴ
11	ش	ﺷ	ﺸ
12	ص	ﺻ	ﺼ
13	ض	ﺿ	ﻀ
14	ط	ﻃ	ﻄ
15	ظ	ﻇ	ﻈ
16	ع	ﻋ	ﻌ
17	غ	ﻏ	ﻐ
18	ف	ﻓ	ﻔ
19	ق	ﻗ	ﻘ
20	ک	ﮐ	ﮑ
21	گ	ﮔ	ﮕ
22	ل	ﻟ	ﻠ
3	م	ﻣ	ﻤ
24	ن	ﻧ	ﻨ
25	ہ	ﮨ	ﮩ
26	ھ	ﮬ	ﮭ
27	ء	-	-
28	ی, ے	ﯾ	ﯿ

Table 4.2: Numbering scheme of characters occuring at Initial and Middle Position.



Character Level Numbering for Final Position Characters	
Character Numbering	Character
1	ا
2	ب
3	پ
4	ت
5	[illegible]
6	ٹ
7	ث
8	ج
9	چ
10	ح
11	خ
12	د
13	ڈ
14	ذ
15	ر
16	ڑ
17	ز
18	س
19	ش
20	ص
21	ض
22	ط
23	ظ
24	ع
25	غ
26	ف
27	ق
28	ک
29	گ
30	ل
31	م
32	ن
33	ں
34	و
35	ؤ
36	ہ
37	ۃ
38	ھ
39	ی
40	ے

Table 4.3: Numbering scheme of characters occuring at Final Position.

4.2.2 Ligature Generation

After giving numeric labels to the characters, all the exhaustive combinations of characters for ligature length of 2 and 3 were generated by using MATLAB. This automated the process of ligature generation.

The ligature's names were formed by joining the character's numbering (as given in the Tables 4.2 and 4.3) with underscore. The way the characters were joined in the ligatures, same was the naming combination of the ligatures. The generic name of a ligature of length 2 and 3 is as follows;

'Ligature Length_ Initial Character_Middle Character_Last Character_0_0_0_**.png'**

The 0s are dynamic and can be replaced as more characters are added in the middle positions. The length of ligatures is dynamic. We have attempted to create ligatures of length 2 and 3 but it is not fixed, rather ligatures of length 4 onwards can also be generated in the same manner. So, the naming scheme used for ligatures is also dynamic and can be extended based on the length of the ligatures.

Ligatures are stored in the form of PNG images. An example showing 5 images with their names for 2 and 3 length ligatures are shown in the figures below;

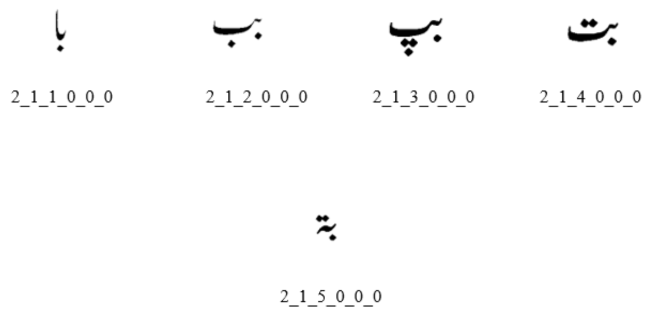


Figure 4.1: Numeric Labels of 2 length ligatures based on character numbers

The first letter 2 is constant in all the above ligatures. This 2 here refers to the ligature length and is same in all the images because these are all 2 character length ligatures. The second number refers to the position of first character and third number is the character number at second position in the ligature.

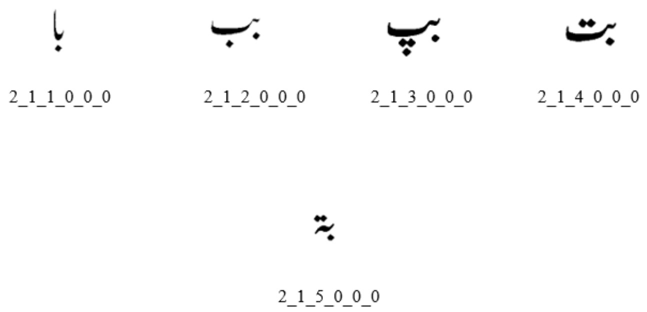


Figure 4.2: Numeric Labels of 3 length ligatures based on character numbers

Similarly in the three length ligature, the first number is 3 in all the images because it refers to the ligature length, which is 3. The later three numbers are the character numbering at the initial, middle and final position of the ligature as referred by the tables 4.2 and 4.3.

4.2.3 Data Augmentation

The dataset at this stage had 1120 ligatures of 2 character length and 31360 ligatures of 3 character length. Forming total 32480 ligatures with only a single image in each class. The classes here refer to each image forming a separate class, as no grouping could occur between ligatures due to their unique shape and properties. Owing to this issue of data scarcity, data augmentation was applied over the images. Each image that formed a separate class was subjected to various augmentations including; These transformations along with the original image created 14

Augmentation Technique	**Number of images generated from each augmentation**
Dilation	3
Erosion	3
Rotation	3
Transformation	3
S & P Noise	1
Original Image	1
	14

Table 4.4: Number of images generated after data augmentation.

images for a single ligature. An example showing all the transformations applied to a single ligature are shown in the Table 4.5

Augmentation Techniques	Images		
Dilation			
Erosion			
Rotation			
Transformation			
S & P Noise			
Original Image			

Table 4.5: Example of application of Augmentation Technique over a single image

The data augmentation method also resulted in re-naming the images to highlight the type of augmentation method applied to the ligature. The name of the ligatures was modified as follows;

'Ligature Length_ Initial Character_Middle Character_Last Character_0_0_0**augmentaion-method.png'**

Rest of the name was same with the addition of the name of the augmentation method written at the end of the ligature name.

This method increased the number of images in each class to 14. The updated number of images in each of the ligature set are shown in the table 4.6.

Ligatures	**Number of Ligature Images after Augmentation**
2 Character Ligatures	15680
3 Character Ligatures	439040
2 and 3 character Ligatures combined	454720

Table 4.6: Number of Images in each Dataset after Data Augmentation

4.2.4 Low Intra-class Variance and High number of classes

The images in each class had increased to a reasonable number which was enough to train a deep learning network. But there was one problem still present with this dataset which was low intra class variability and high number of classes. The updated dataset had 32480 classes with 14 images in each class. The dataset had large number of classes but the number of samples within each class was very small. Such a dataset when used to train a deep learning model would have caused the model to have high bias and low variance causing the model to overfit. An overfitted model has high training accuracy but low test data accuracy. Such a model does not perform well on unseen data producing low performance deep learning network. There was a need to solve this issue before moving on to the next phase of the classification problem.

4.2.5 Redefining classes

To overcome the problem of low intra-class variability and high number of classes we proposed a unique method to reduce the number of classes and to increase the number of samples within each class. According to the proposed technique, the ligatures were grouped according to the similarity between their primary components. A structural analysis of the ligatures shows that many of the ligatures differed only in their secondary components which are the dots and the diacritic marks while the rest of the basic structure of the ligature was exactly same. Owing to this property, we grouped together ligatures based on same primary component.

An example of this is shown in table 4.7 where bay−alif, pay−alif, tay−alif, te−alif, say−alif, noon−alif, choti−hay-alif, hamza−alif all have the same primary component. Looking at this example, the 8 different classes were merged into a single class based on the similarity between their primary component. The

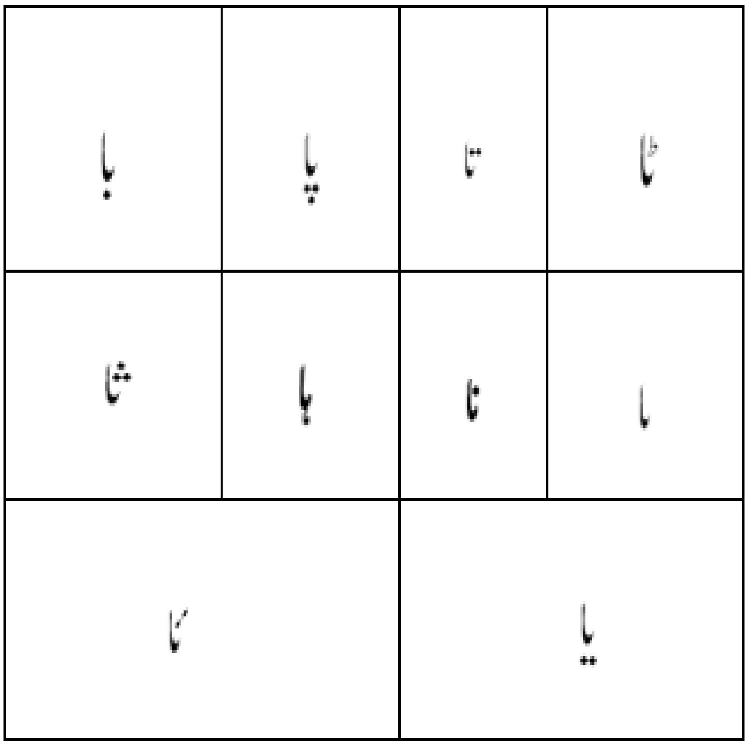


Table 4.7: Combining 8 various ligatures into a single class based on their similar primary component.

primary component of all the above shown ligatures is same and is displayed in the following figure 4.3

Figure 4.3: Combining 8 various ligatures into a single class based on their similar primary component

For this specific class, a total of 10 previous classes having 14 images in each class were grouped together into one single class. This makes this single class to have 140 images.

A complete set of images in a class are shown Figure 3.3, which represents images from a single class where the images are grouped based on their primary structural similarity. This method when applied to all the other classes reduced

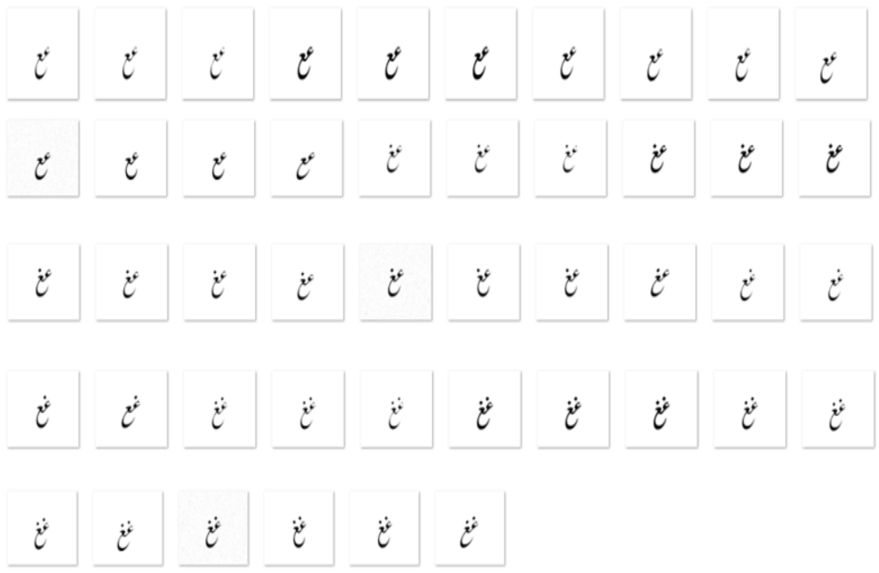


Figure 4.4: Image samples of the class 'ain−ain' formed after structure based Classification

the number of classes from 32480 to 3116 classes only, for both 2 and 3 character length ligatures combined. This method greatly reduced the number of classes and increased the number of samples within each class.

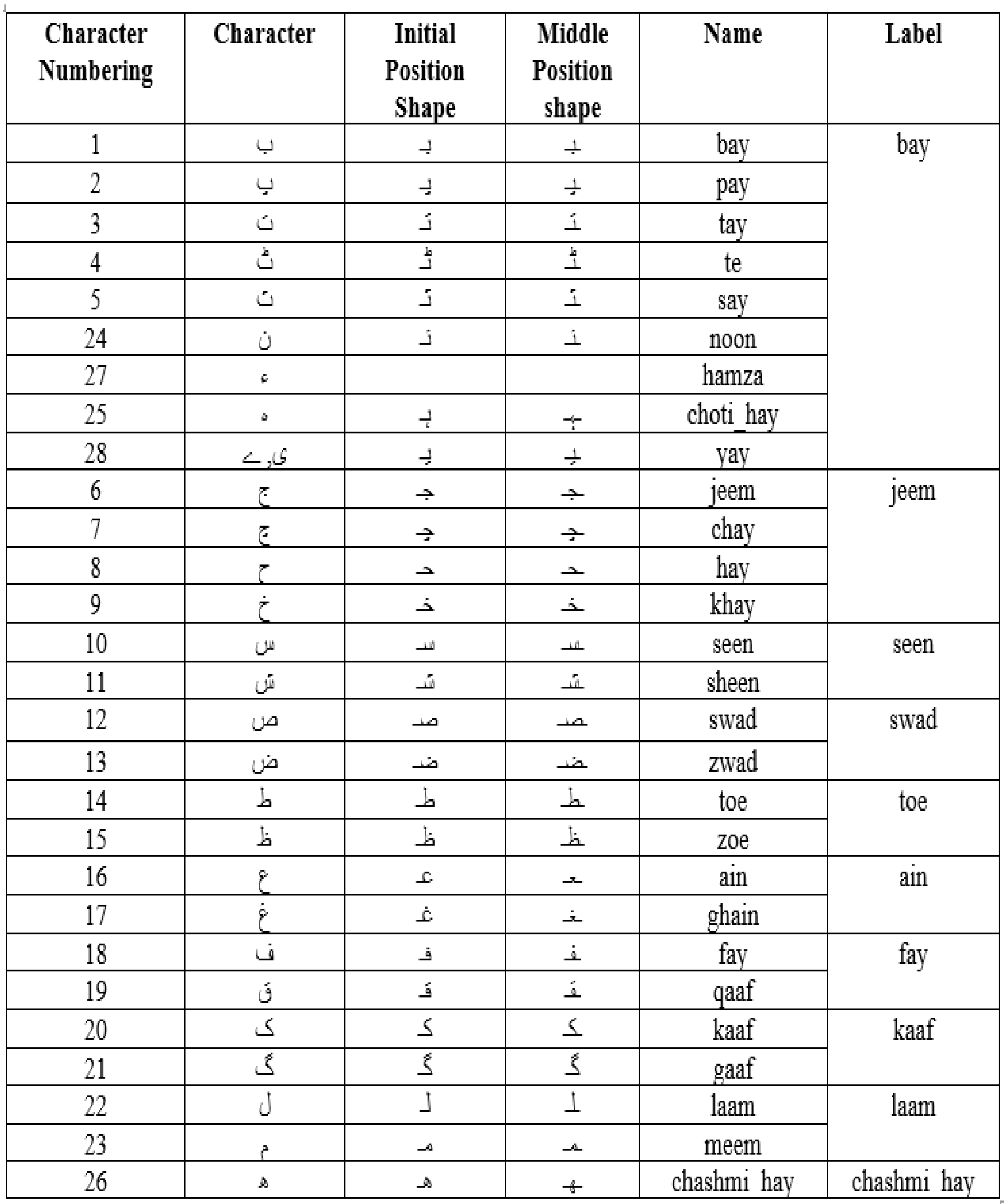
Ligature Set	Number of Classes
2 character ligature	236
3 character ligature	2880
2 and 3 character ligature combined (Final Dataset)	3116

Table 4.8: Number of classes in the ligature set after redefinition of classes

4.2.6 Re-Labelling Ligatures

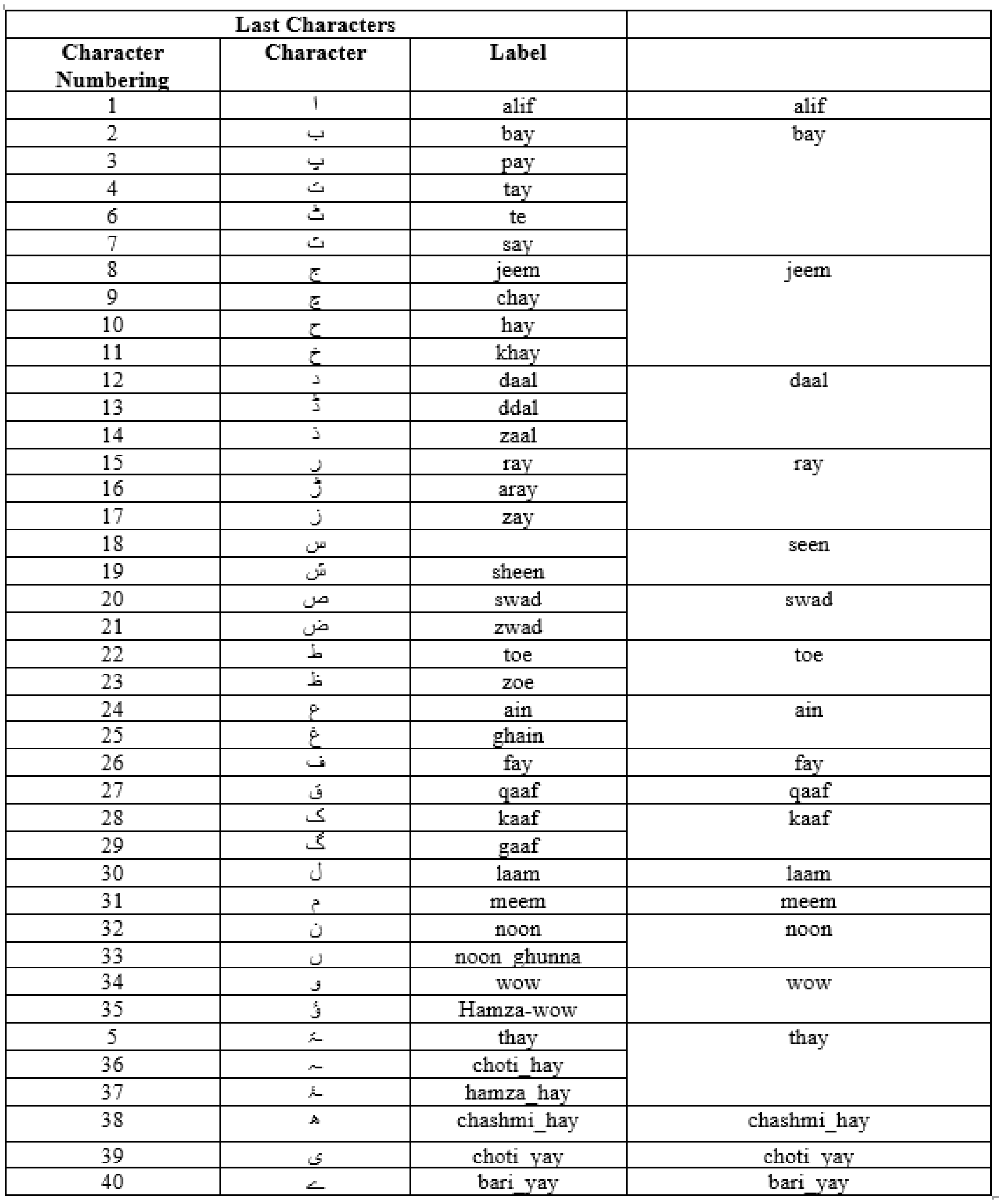
The ligatures needed to be regrouped based on their structural similarity. The number of ligatures was 454720. Manual grouping of such a large number of ligatures was a very time taking task. To automate this process, the ligatures were re-labelled. This re-labelling was done at the character level. As we had seen in the section 4.1, that the initial numeric labels of the ligatures were formed by giving numbers to the characters, this time those characters were grouped based on similar primary component and after combining they were given a new name. Each character already had a number associated with it. The characters whose primary components were same, they were grouped together and given one label. The new label was based on the name of the character with the lowest number in that group This time, more realistic labels were given to the characters, instead of giving numeric labels we relabeled the grouped characters by names as shown in the following table.

The number of characters that occur at initial and middle position in a ligature reduced from 28 to 11.



Character Numbering	Character	Initial Position Shape	Middle Position shape	Name	Label
1	ب	بـ	ـبـ	bay	bay
2	پ	پـ	ـپـ	pay	
3	ت	تـ	ـتـ	tay	
4	ٹ	ٹـ	ـٹـ	te	
5	ث	ثـ	ـثـ	say	
24	ن	نـ	ـنـ	noon	
27	ء			hamza	
25	ہ	ہـ	ـہـ	choti_hay	
28	ی, ے	یـ	ـیـ	yay	
6	ج	جـ	ـجـ	jeem	jeem
7	چ	چـ	ـچـ	chay	
8	ح	حـ	ـحـ	hay	
9	خ	خـ	ـخـ	khay	
10	س	سـ	ـسـ	seen	seen
11	ش	شـ	ـشـ	sheen	
12	ص	صـ	ـصـ	swad	swad
13	ض	ضـ	ـضـ	zwad	
14	ط	طـ	ـطـ	toe	toe
15	ظ	ظـ	ـظـ	zoe	
16	ع	عـ	ـعـ	ain	ain
17	غ	غـ	ـغـ	ghain	
18	ف	فـ	ـفـ	fay	fay
19	ق	قـ	ـقـ	qaaf	
20	ک	کـ	ـکـ	kaaf	kaaf
21	گ	گـ	ـگـ	gaaf	
22	ل	لـ	ـلـ	laam	laam
23	م	مـ	ـمـ	meem	
26	ھ	ھـ	ـھـ	chashmi_hay	chashmi_hay

Table 4.9: Re-labelling of characters based on structural similarity of characters at initial and middle position within a ligature.



Last Characters			
Character Numbering	**Character**	**Label**	
1	ا	alif	alif
2	ب	bay	bay
3	پ	pay	
4	ت	tay	
6	ٹ	te	
7	ث	say	
8	ج	jeem	jeem
9	چ	chay	
10	ح	hay	
11	خ	khay	
12	د	daal	daal
13	ڈ	ddal	
14	ذ	zaal	
15	ر	ray	ray
16	ڑ	aray	
17	ز	zay	
18	س		seen
19	ش	sheen	
20	ص	swad	swad
21	ض	zwad	
22	ط	toe	toe
23	ظ	zoe	
24	ع	ain	ain
25	غ	ghain	
26	ف	fay	fay
27	ق	qaaf	qaaf
28	ک	kaaf	kaaf
29	گ	gaaf	
30	ل	laam	laam
31	م	meem	meem
32	ن	noon	noon
33	ں	noon_ghunna	
34	و	wow	wow
35	ؤ	Hamza-wow	
5	ـۃ	thay	thay
36	ـہ	choti_hay	
37	ـۂ	hamza_hay	
38	ھ	chashmi_hay	chashmi_hay
39	ی	choti_yay	choti_yay
40	ے	bari_yay	bari_yay

Table 4.10: Re-labelling of characters based on structural similarity of characters at Final position within a ligature.

Similarly, the number of characters at final position reduced from 40 to 20 characters only.

It must be considered that these reductions in the number of characters is based on their structure and accompanies to the similarity in the primary component at the ligature level.

This process led us to have the final dataset ready to go for the next phases of classification. This dataset had total of 454720 images grouped into 3116 classes for both 2 and 3 character length ligatures.

4.2.7 Summary Statistics of the Dataset

Ligature Set	Min images	Max images	Avg images
2 Character Ligature Set	14	560	66
3 Character Ligature Set	14	4480	153
(Final Dataset)	14	4480	145

Table 4.11: Summary of the number of images in the Urdu Ligature Dataset

The dataset contained data in three forms;

- 2 Character Length Ligatures
- 3 Character Length Ligatures
- 2 and 3 Character Length Ligatures Combined (Final Dataset)

The Resnet34 model was trained separately for all the three sets of the data. The reason to do this was to identify the classification accuracies in all the three cases and to determine if any difference occurs by training 2 and 3 character data in isolation as well as combined. The combined ligature set is named as the Final Dataset. In the next phases of classification, the results will be represented separately for all the three cases of the datasets.

4.3 Pre-Processing

The dataset created for Urdu ligatures had images in binary form. These images were grey scale and therefore did not require RGB to Grey Scale conversion. The following pre-processing operations were performed;

Image Resizing: The images were of size 600x500. The size of the images was very large and training the network using this size of images would be highly computational complex taking a lot of training time. To save from this time complexity, the images were resized to 224x224.

Random Cropping: The images were taken as randomly cropped for training.

By random cropping, it does not mean the loss of any part of the image, rather the images are trained by recursively cropping a random area and use that area to learn features. This way all the various positions of the image is used to feature extraction.

Random Augmentation: Random Augmentations were also applied in batches to learn various forms of writing styles in which the ligature can occur in printed format.

The following figures show the single batch of images after pre-processing for 2, 3 character ligatures and the Final Dataset respectively.

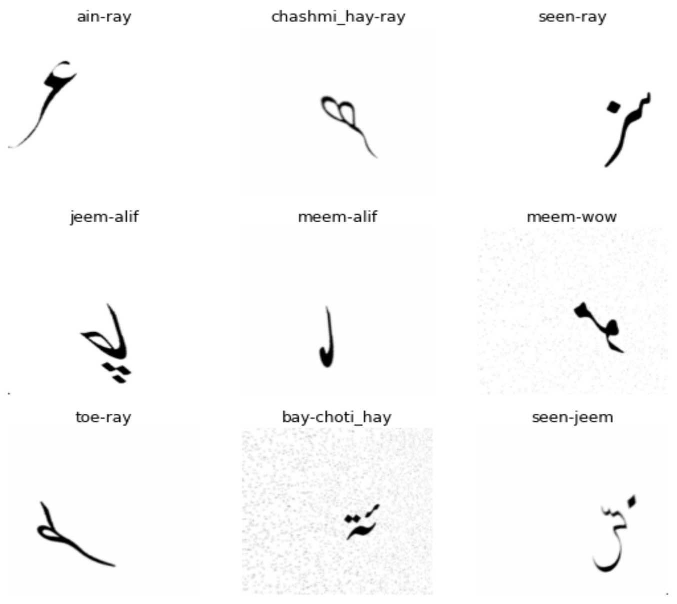


Figure 4.5: A single batch of ligature images after pre-processing (2 Character Ligature Set)

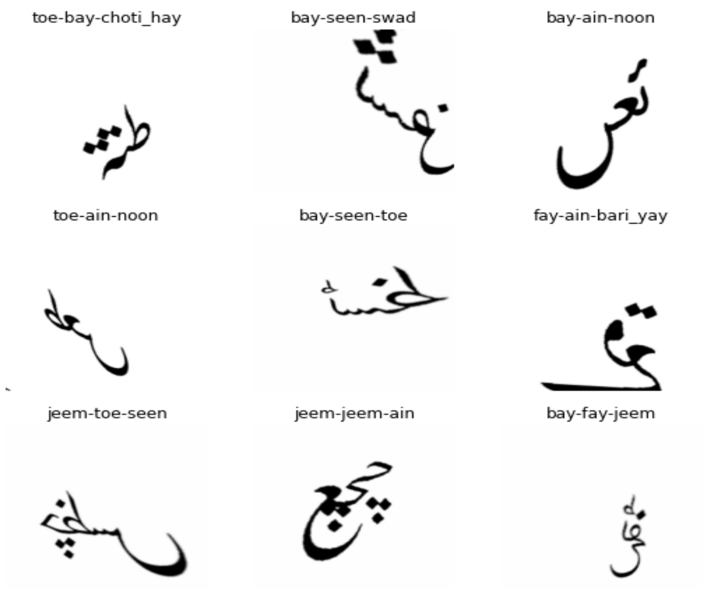


Figure 4.6: A single batch of ligature images after pre-processing (3 Character Ligature Set)

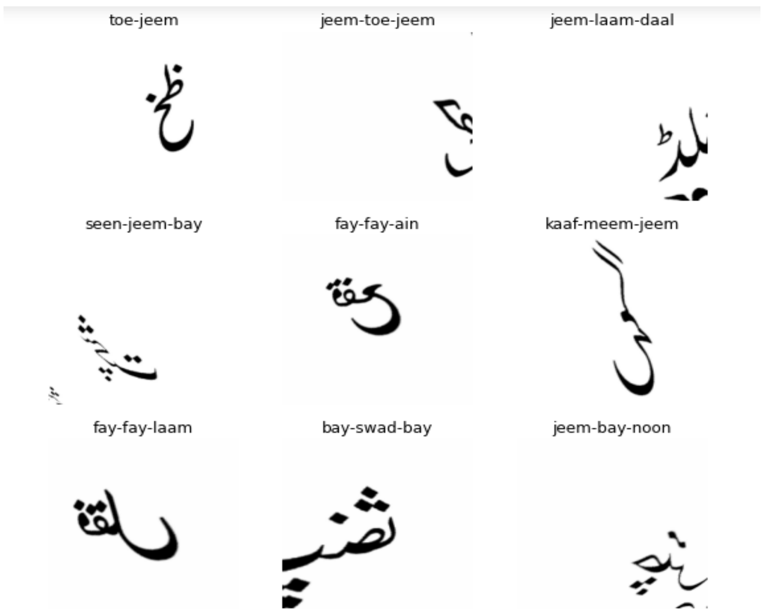


Figure 4.7: A single batch of ligature images after pre-processing (2 and 3 Character Ligature Combined)

4.4 Training Model

The training model used for ligature classification was Resnet34. It is a 34 layer architecture network with short connections. The network uses feature learning to speed up the training process. This model is already trained on ImageNet dataset. We imported the network using cnn_learner. The training data set was randomly split into 80/20. 80% of the data was used for training and the rest was used as the validation data.

4.5 Hyperparameter Tuning

The Resnet34 model was a pre-trained network. During the training phase the weights of the convolutional layers were to be updated. So, before starting off with the training process it was very important to fine tune the hyperparameters. Hyperparameter tuning is the process to find the appropriate values of hyperparameters of a deep learning or machine learning algorithm. Hyperparameter settings have a direct impact on the performance of a training network. These hyperparameter settings suggest how long an algorithm will take to converge or train. Wrong hyperparameter settings can lead to poor results of the deep learning model [55].

Before starting off with the training of the model over our dataset, the optimal learning rate of the dataset was found. This method helped to find out the optimal learning rate for our experiment. The graph showing the optimal learning rate as calculated for training of our Urdu ligature dataset for all the three cases is shown in the Figure 4.8, 4.9 and 4.10.

```
SuggestedLRs(valley=0.0030199517495930195)
```

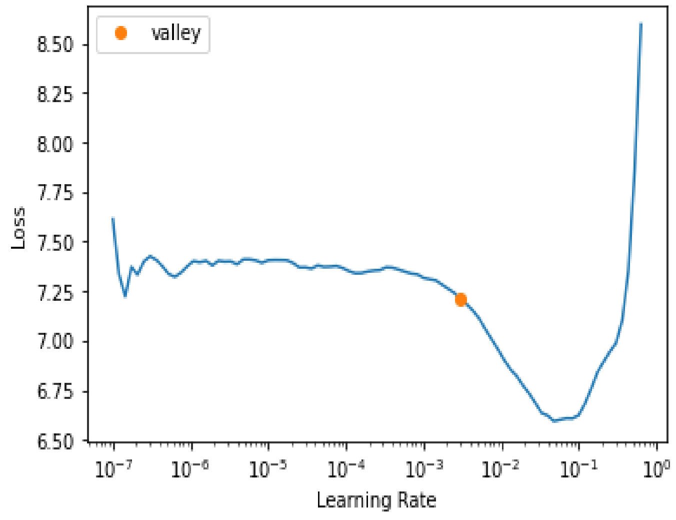


Figure 4.8: Graph representing the optimal learning rate for 2 Character Ligature Set

```
SuggestedLRs(valley=0.0025118864141404063)
```

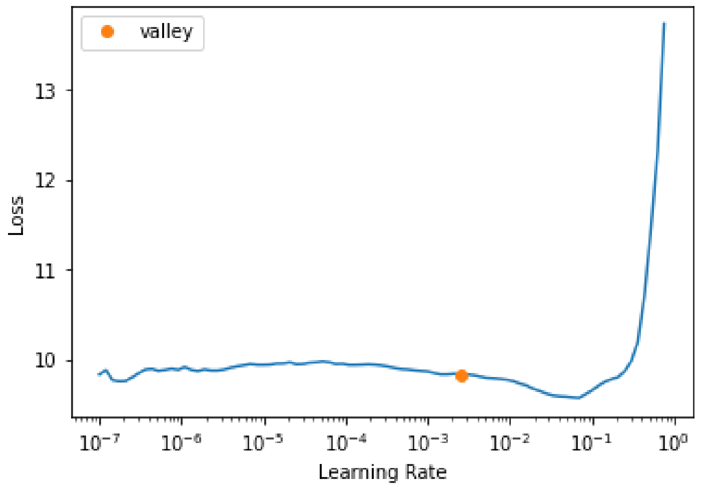


Figure 4.9: Graph representing the optimal learning rate for 3 Character Ligature Set

```
SuggestedLRs(valley=0.0063095735386013985)
```

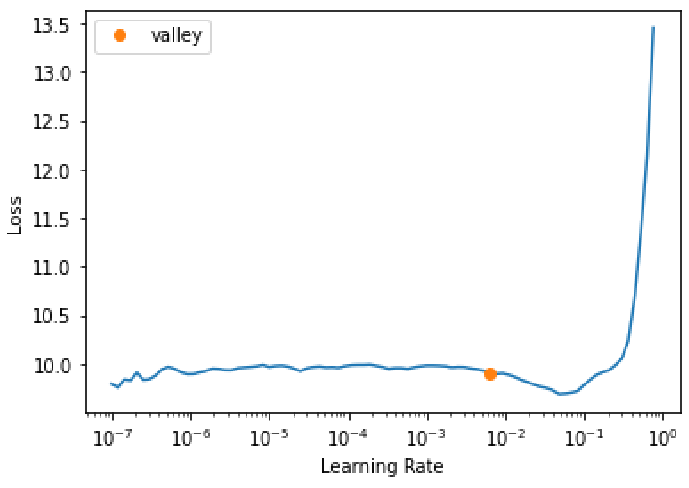


Figure 4.10: Graph representing the optimal learning rate for combined 2 and 3 Character Ligature Set

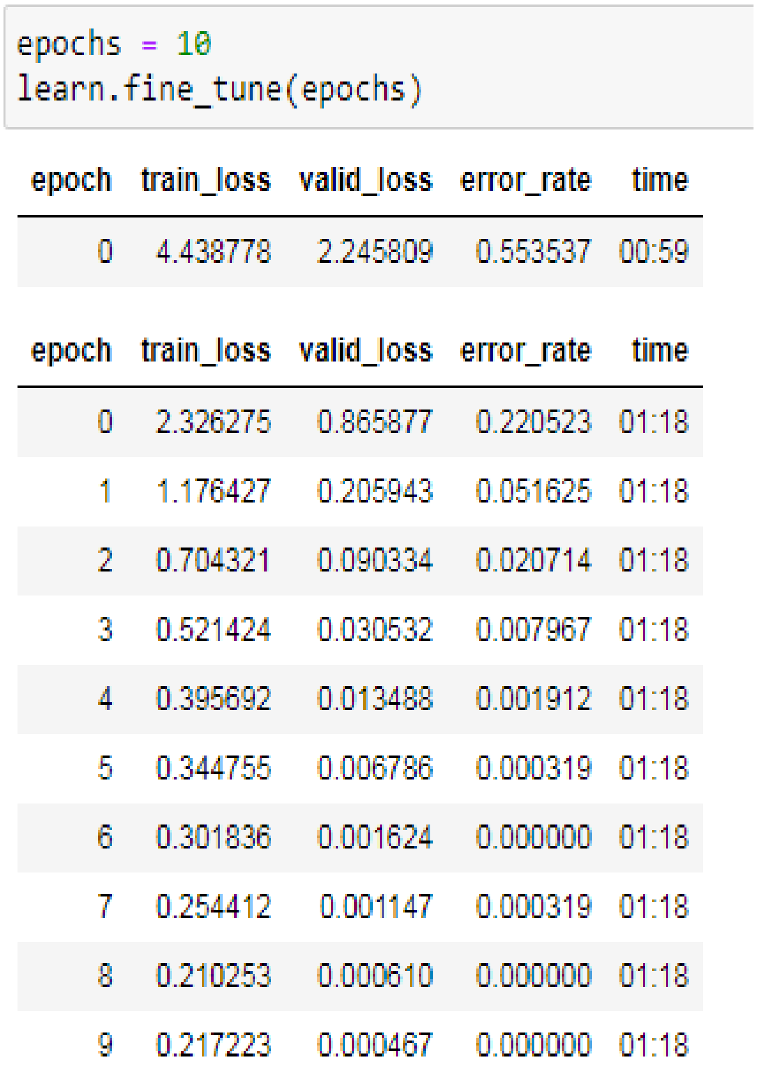
These learning rates were automatically calculated. The graphs represent the learning rate in comparison with the training loss on the dataset. The loss is calculated against every learning rate. The blue line represents the trend whereas the dot represents the valley. This point represents the optimal value of learning rate that will give the minimum loss.

4.6 Training of the Resnet34 Network

Three networks were trained separately for all the three datasets which were 2 character ligature set, 3 character ligature set and the ‘Final Dataset’ that had all the combinations of 2 and three character sets combined. The number of epochs was set to 10. Three model were trained and the train loss, validation loss and error rate were calculated at each epoch. The details of the training for all the three cases are shown in the figures below;

4.6.1 Training network for 2 Character Ligature Set

The figure below shows the result of training the Resnet34 model over the 2 character ligature set.
Figure 4.11 shows the outcome of training the network for classification of 2 char-



```
epochs = 10
learn.fine_tune(epochs)
```

epoch	train_loss	valid_loss	error_rate	time
0	4.438778	2.245809	0.553537	00:59

epoch	train_loss	valid_loss	error_rate	time
0	2.326275	0.865877	0.220523	01:18
1	1.176427	0.205943	0.051625	01:18
2	0.704321	0.090334	0.020714	01:18
3	0.521424	0.030532	0.007967	01:18
4	0.395692	0.013488	0.001912	01:18
5	0.344755	0.006786	0.000319	01:18
6	0.301836	0.001624	0.000000	01:18
7	0.254412	0.001147	0.000319	01:18
8	0.210253	0.000610	0.000000	01:18
9	0.217223	0.000467	0.000000	01:18

Figure 4.11: Training results of 2 Character Ligature Set

acter ligature set after each epoch. train_loss is the loss calculation on the train data which accounted for 80% of the ligatures. Similarly valid_loss is the loss calculation on the validation data accounting for 20% of the 2 character ligatures. error_rate is the rate with which the training model wrongly classifies a ligature [8]. The error rate at the first epoch is 0.55 which is a quite high value. The error rate decreases to 0.22 in the next epoch and gradually in each epoch it keeps on decreasing and goes to 0 in the final epoch. The graph representing the trend of train and validation loss is shown in Figure 4.12.

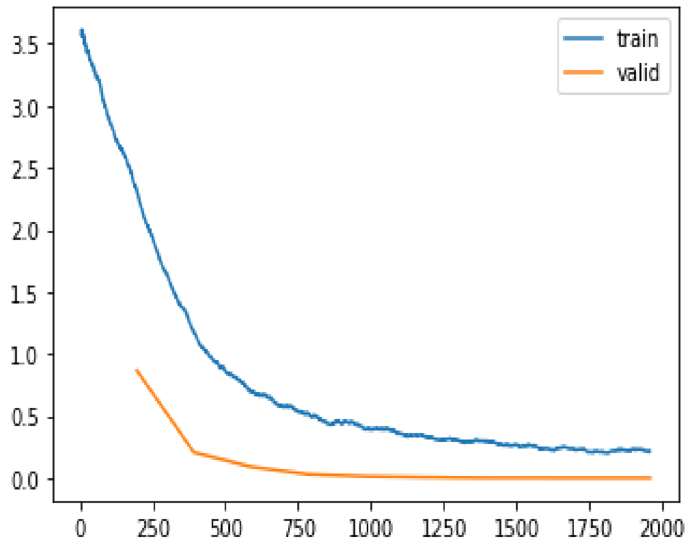


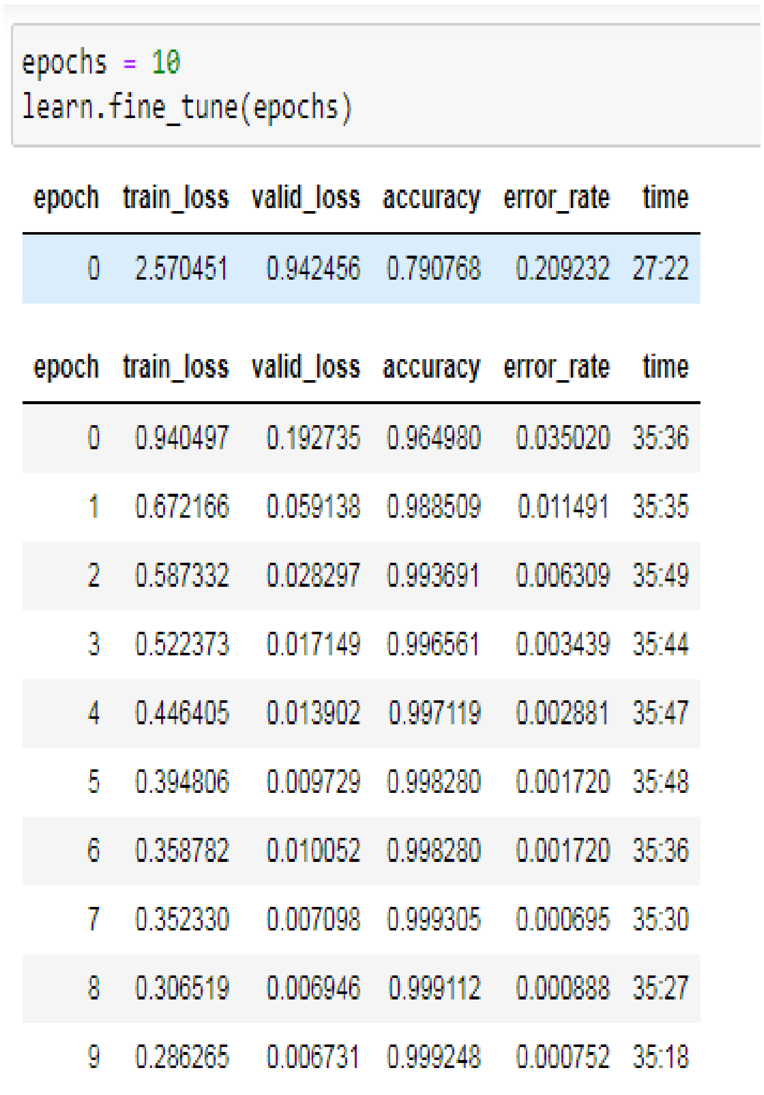
Figure 4.12: Graph representing the trend of train loss to validation loss for 2 character ligature set

The graph shows a comparison of the rate of decrement of train and validation losses of the 2 character ligature dataset. The x-axis shows time whereas the y-axis represents the loss. Focusing on the blue line representing the train loss, initially the train loss was very high which was more than 3.5, with time the loss kept on decreasing till it reached almost to 0.

Similarly, the orange curve representing the validation loss has also the same trend and kept on decreasing till it reached 0. It is very important to note that both the train and validation losses decreased with time which indicates a high performing training model because on one hand the train loss is decreasing and on the other hand the loss on the validation data is also decreasing. If this would not have been the case and the validation loss would have increased with time, it would have indicated an overfitted model which shows high train data accuracy and low validation set accuracy. Whereas in this case, the validation loss was also decreasing indicating good performance of the model on the validation data as well as the train data.

4.6.2 Training network for 3 Character Ligature Set

The Resnet34 model was also trained over the 3 character ligature set. The results of training are; The metrics calculated at each epoch are train loss, validation loss, accuracy and error rate.



```
epochs = 10
learn.fine_tune(epochs)
```

epoch	train_loss	valid_loss	accuracy	error_rate	time
0	2.570451	0.942456	0.790768	0.209232	27:22

epoch	train_loss	valid_loss	accuracy	error_rate	time
0	0.940497	0.192735	0.964980	0.035020	35:36
1	0.672166	0.059138	0.988509	0.011491	35:35
2	0.587332	0.028297	0.993691	0.006309	35:49
3	0.522373	0.017149	0.996561	0.003439	35:44
4	0.446405	0.013902	0.997119	0.002881	35:47
5	0.394806	0.009729	0.998280	0.001720	35:48
6	0.358782	0.010052	0.998280	0.001720	35:36
7	0.352330	0.007098	0.999305	0.000695	35:30
8	0.306519	0.006946	0.999112	0.000888	35:27
9	0.286265	0.006731	0.999248	0.000752	35:18

Figure 4.13: Training results of 3 Character Ligature Set

Train loss: The train loss was 2.57 in the first epoch and gradually decreased in every epoch and was 0.2 in the last epoch. This trend shows a decrease in the loss on the train data for three character ligature set. Decreasing loss indicates that the model was learning the features better over the train data after each epoch.
Validation loss: The validation loss was at a high value of 0.94 in the first epoch, 0.19 in the second epoch and kept decreasing and reached to 0.006 in the final epoch. This decrease indicated that the loss was lowering on the validation data with each epoch meaning that the model was trained well to predict the validation

data ligatures correctly.
Accuracy: This metric represents how well the model is predicting over the validation data. The accuracy at the first epoch was 0.79, 0.96 in the next epoch and 0.99 in the next epochs. The accuracy shows an increasing trend with each epoch. This shows that model was learning better in each epoch and correctly classified the validation data ligatures.
Error rate: The error rate again showed a decreasing trend. It was high with an initial value of 0.2 and decreased gradually in each epoch to 0. The decreasing error rate showed increasing performance of the network over the validation data. The trend of validation and train losses is shown with time. The graph indicates

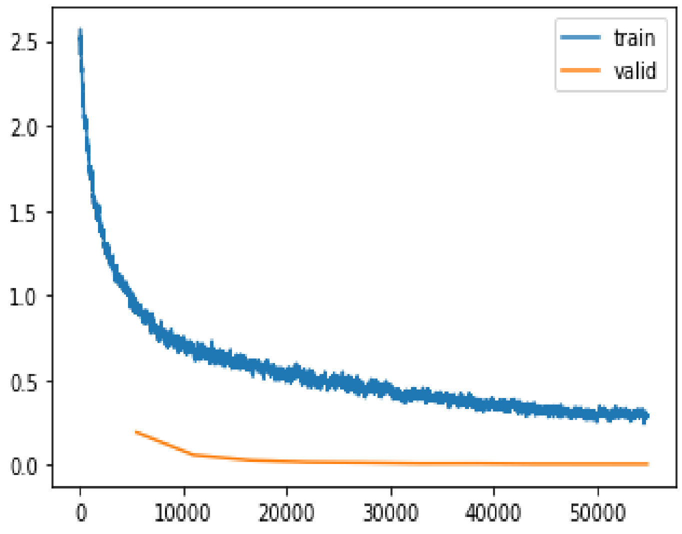


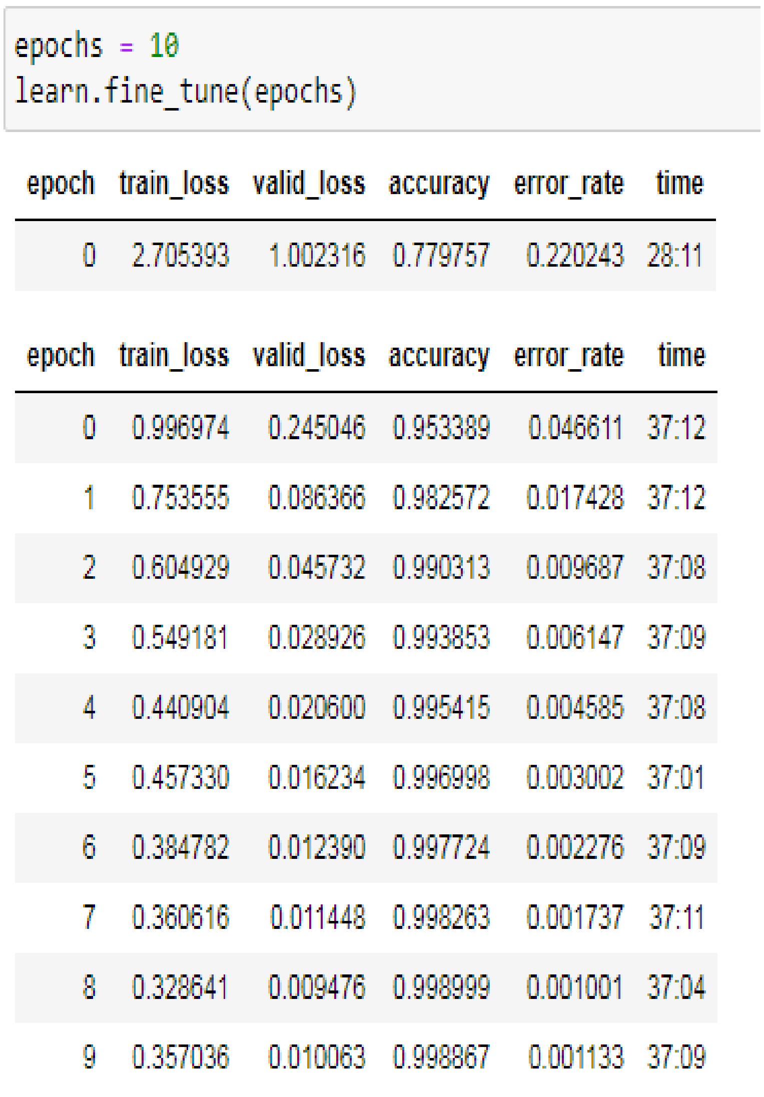
Figure 4.14: Graph representing the trend of train loss to validation loss for 3 character ligature set

that the train loss decreased quickly at the start and later slowly after reaching a 0.5 error rate. After reaching 0.5 the loss decreased slowly over the train data with time and ended at 0.2 in the last epoch.

The validation loss was not very high even at the start and gradually decreased in each epoch to 0. The above trend indicates a high performance trained model, as it showed that the train loss did not reach to 0 at the end whereas the validation loss became 0. This indicates that the model was not overfitted rather it showed a suitable train loss trend and an excellent validation loss trend. A model whose validation accuracy is high, and loss is low indicates that the model can predict the data well when given unseen data as it can predict well on the validation data.

4.6.3 Training Network for combined 2 and 3 Character Ligature Set

The last model trained was over the dataset that had combined ligatures from the 2 and 3character ligature set. This model was a big task for training as the number of images was very high. There were 454720 images to be trained. This dataset was 96% larger than the 2 character ligature set and 10% larger than the three character ligature set. It had 3116 classes. The training results over this dataset on the Resnet34 model were as follows;



```
epochs = 10
learn.fine_tune(epochs)
```

epoch	train_loss	valid_loss	accuracy	error_rate	time
0	2.705393	1.002316	0.779757	0.220243	28:11

epoch	train_loss	valid_loss	accuracy	error_rate	time
0	0.996974	0.245046	0.953389	0.046611	37:12
1	0.753555	0.086366	0.982572	0.017428	37:12
2	0.604929	0.045732	0.990313	0.009687	37:08
3	0.549181	0.028926	0.993853	0.006147	37:09
4	0.440904	0.020600	0.995415	0.004585	37:08
5	0.457330	0.016234	0.996998	0.003002	37:01
6	0.384782	0.012390	0.997724	0.002276	37:09
7	0.360616	0.011448	0.998263	0.001737	37:11
8	0.328641	0.009476	0.998999	0.001001	37:04
9	0.357036	0.010063	0.998867	0.001133	37:09

Figure 4.15: Training results of combined 2 and 3 Character Ligature Set

Train loss: The results of network training indicate that the train loss was very high in the first epoch i.e., 2.7 then it decreased to 0.99 in the next epoch and kept on decreasing till it reached 0.35 in the last epoch. This decreasing trend as indicated earlier for 2 and 3 character ligature set indicated that the model was learning features gradually with each epoch and was getting better. The train loss did not end up at 0 which indicated that the model was not overfitting, because otherwise it would have given 100% accuracy of prediction on the train data.
Validation loss: The validation loss shows a very rapid decrease with each epoch.

It had a value of 1.0 in the first epoch ,0.2 in the second epoch and kept on decreasing to each epoch to 0.01 in the last epoch. This trend indicated increasing classification accuracy over the validation data.
Accuracy: The accuracy on the validation data was lower that is 0.77 in the first epoch and then rapidly increased to 0.95 in the next epoch, 0.98 in the third epoch, and 0.99 in the fourth epoch and after that it remained around 0.99 in the next epochs. The high accuracy of the model indicates a very high performance of the trained model as it can predict on the validation data accurately.
Error rate: Error rate is the opposite of accuracy. It was high initially at 0.22, then decreased to 0.04 in the next epoch and kept on decreasing in each epoch till it reached to 0.001 in the last epoch.

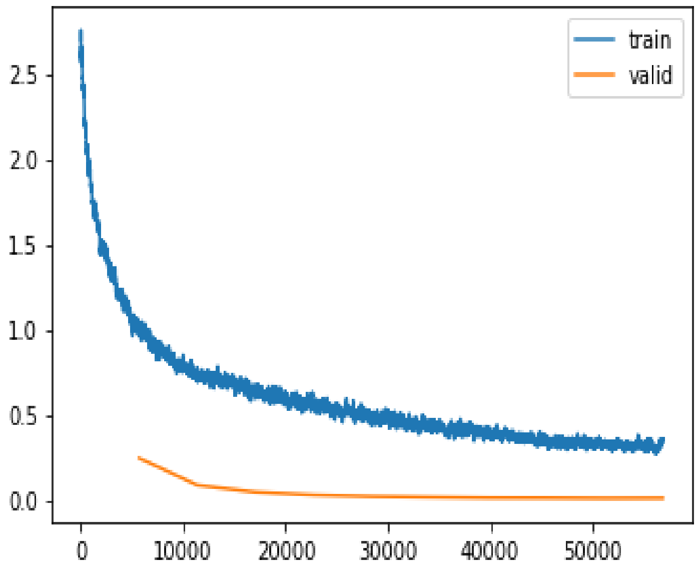


Figure 4.16: Graph representing the trend of train loss to validation loss for combined 2 and 3 character ligature set

The graphical trend of the train and validation loss can be seen in the figure 4.16. It is the representation of the trend of train loss to validation loss. The train loss decreased quickly from 2.5 to 1.0 in the start and then decreased gradually with time.

The validation loss decreased quickly in the start and after reaching close to 0 remained constant. The train loss did not completely become 0. It is an indication that the model was not overfitting rather a slight loss was present so that the model does not overfit to train data and is flexible to predict well over the unseen data.

The graph also indicated the convergence between the train and validation loss curves. The train loss and validation loss both were decreasing with each epoch, but the trend of train loss was not very sharp and did not completely become 0. If it would have become 0 completely this would indicate no loss at all meaning an overfitted model whereas if the value would have been high, it would have indicated underfitted model which has not learned the features well. There should always be a suitable balance of ratio and the values should not be too high or too low rather they should be close to the lower side and that was the result of training over our dataset. The train loss was lower but not 0 meaning the model was neither overfit nor underfit. Similarly, the low validation loss which reached to 0.01 indicate a very high performance over the validation data.

This high performance over the validation data indicated that the model would also produce good results on the test data. The model is trained well and can predict well on unknown data as well. To evaluate the performance of our trained model, the next step was to test it over the unseen data.

4.7 Test Data

The test data is required to evaluate the model to identify how well the trained model can predict over the unseen data. It is a very vital step in any deep learning classification task because otherwise the generalization capability of the model cannot be calculated [3].

We had trained three separate models for the prediction of all the three ligature sets, which were 2 character ligature set, 3 character ligature set and final dataset which comprised both the previous ligature set images combined. So, we also had to create three test sets which included 20% images from the respective datasets.

4.7.1 Acquiring Test Data Images

The first step in the generation of test data was to obtain 20% random images from the dataset. Three test sets were generated from the three datasets. The number of images in each of the test sets were;

Ligature Sets	**Number of Images**
2 Character Ligature Set	3138
3 Character Ligature Set	87807
(2 & 3 Character Ligature Set Combined) Final Test Set	90,945

Table 4.12: Number of Images in each Test Set

4.7.2 Update Test Data Images

The 20% of the images obtained from the three datasets were not used raw for the training purpose. Rather they were updated by using the data augmentation technique in order to create unseen images for testing the trained network.

Each image in the test data was given a transformation chosen randomly from the following 7 augmentation techniques;

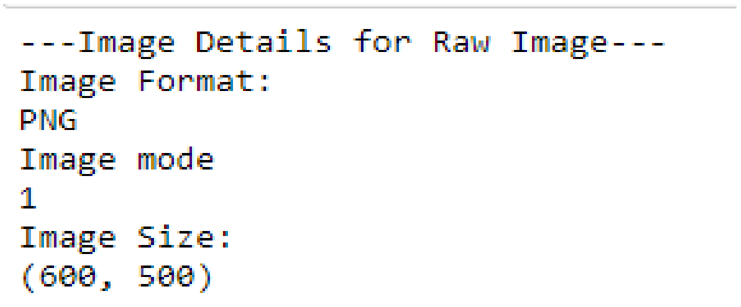
1. Rotation
2. Zoom
3. Width shift
4. Height shift
5. Shear
6. Noise
7. Brightness

The choice of the augmentation was completely random and moreover the range to which each image was subjected to a particular augmentation was also random, to omit any kind of biasness in the test data generation.

4.7.3 Resizing and Random Cropping

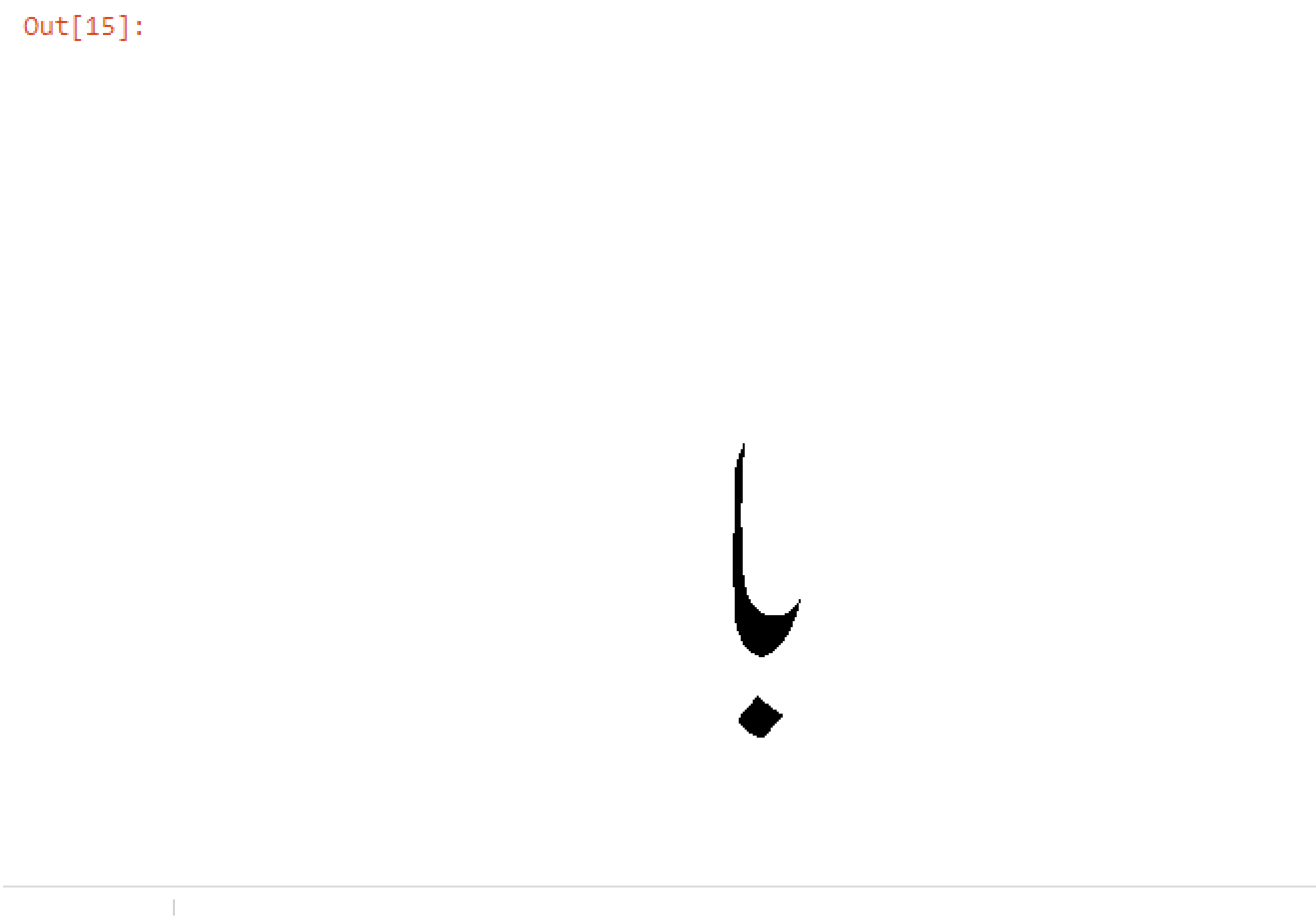
After obtaining the updated test data images for each of the test set. The next step was to preprocess the images in the same way the train data sets were pre-processed as indicated in the section 3.5.2.

Each image was resized to (224,224) and random cropping was applied along with its conversion to binary image. An example showing the image ‘2_1_1_0_0_0dilation_3.png’, before and after pre-processing is shown in the following figures:



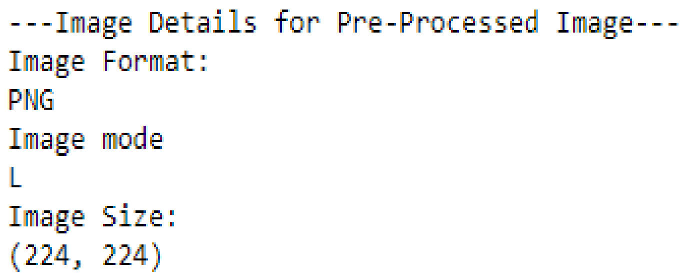
```
---Image Details for Raw Image---
Image Format:
PNG
Image mode
1
Image Size:
(600, 500)
```

Figure 4.17: Image Summary for Raw Image after random augmentation



```
Out[15]:
```

Figure 4.18: Raw Image after augmentation



```
---Image Details for Pre-Processed Image---
Image Format:
PNG
Image mode
L
Image Size:
(224, 224)
```

Figure 4.19: Image Summary after Pre-Processing

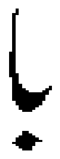


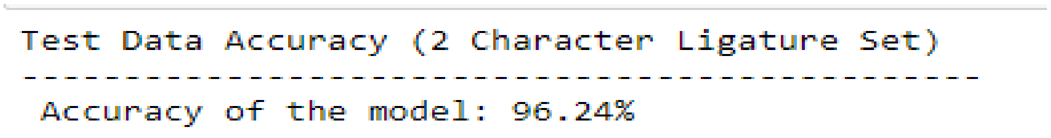
Figure 4.20: Image after Pre-Processing

4.7.4 Predictions on Test Set:

The test datasets finally generated for all the three sets were tested over the trained model. Accuracy of the model over the test set for 2, 3 and 2 3 character ligature sets combined is as follows:

Test Data Accuracy for 2 Character Ligature Test Set:

The output of the model is shown in Figure 4.21.



```
Test Data Accuracy (2 Character Ligature Set)
---------------------------------------------
 Accuracy of the model: 96.24%
```

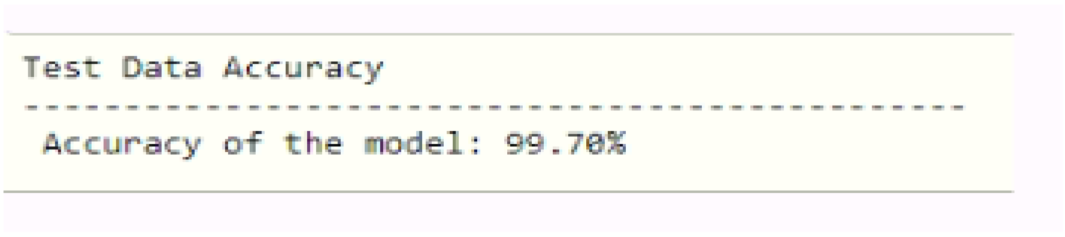
Figure 4.21: Test Data Accuracy for 2 Character Ligature Set.

The test data showed an accuracy of 96.24% over the 2 character ligature trained Resnet34 model.

Test Data Accuracy for 3 Character Ligature Test Set:

The output of the model is shown in Figure 4.22.

The test data for 3 character ligature set was predicted with a remarkable accuracy of 99.7%.

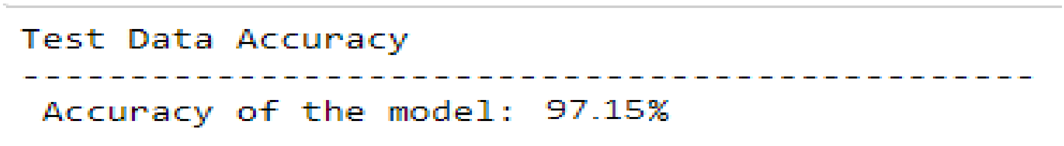


```
Test Data Accuracy
-------------------------------------------------
 Accuracy of the model: 99.70%
```

Figure 4.22: Test Data Accuracy for 3 Character Ligature Set.

Test Data Accuracy for 2 & 3 Character Ligature Set:

The test data for combined 2 and 3 character ligature set gave an accuracy of 97.15% over the network trained over the final dataset (containing the ligatures from the 2 and 3 character set combined).



```
Test Data Accuracy
-------------------------------------------------
 Accuracy of the model:  97.15%
```

Figure 4.23: Test Data Accuracy for combined 2 and 3 Character Ligature Set.

4.8 Summary

This section explains the results of training a deep learning model over the generated dataset. The Urdu ligature dataset contained exhaustive combinations of Urdu characters of length 2 and 3. The dataset was divided into three sets namely 2 character ligature set, three character ligature set and the final dataset which contained the datasets contained in both 2 and 3 character ligature sets. Resnet34 model was used as the deep learning classification model. The results at each step of the classification process are stated as well as explained. Our generated dataset was unique to all other Urdu classification datasets. The large number of ligature images was a big challenge for classification. Not only the dataset had large number of classes with low intra-class variability but also handling the large number of images was a difficult task. We proposed a unique method to handle the challenges present in our generated Urdu ligature dataset. This method has given remarkable validation and test data accuracies. The next step is to evaluate the results and compare them with the work already done over the Urdu ligature datasets. This evaluation and comparison of the proposed technique is done in the next section. □

Chapter 5

Evaluation and Discussion

The previous chapter dealt with showing and explaining the results of the experiments performed over the Urdu ligature dataset. This chapter deals with the evaluation of these results. The results at training, validation and testing would be evaluated for performance. Along with the comparison of the results obtained by the proposed technique would be compared to the existing deep learning methods for Urdu ligature recognition.

5.1 Evaluation of 2 Character Model

The 2 character ligature set comprised of 15680 ligatures which were divided into train and validation data as follows;

2 character Ligature Set	**Ligatures**
Train Data	12544
Validation Data	3136

Table 5.1: Ligatures division in 2 character ligature set

5.1.1 Training and Validation Loss

The training of the Resnet34 was done using 12544 ligatures selected randomly. A total of 10 epochs were trained and the objective of the training was to decrease and validation loss with each epoch. But by the end of the training the train loss should not be lower than the validation loss because that would create an overfitted

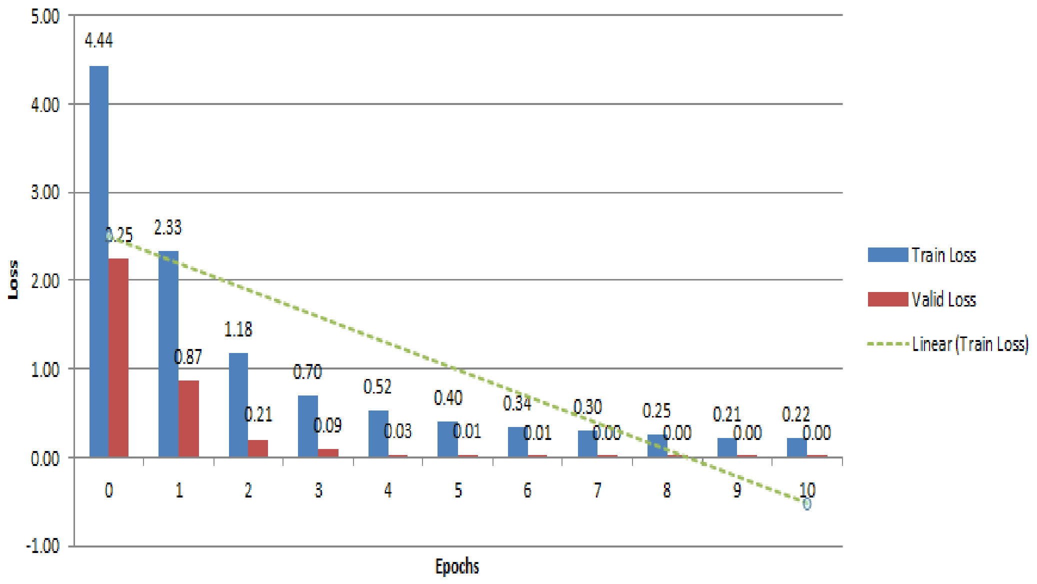


Figure 5.1: Train and Validation Loss at each epoch (2 Character Ligature Set).

model. The graph below shows the training and validation loss at each epoch. The blue bars represent the train loss and red, the validation loss. Initially the train loss was at a very high value of 4.44 at epoch 0. The validation loss was also high but not as much as the train loss and it was 0.25.

With each epoch the train loss decreased and was 0.22 in the last epoch. Whereas the validation loss decreased from 2.25 to 0.87 and kept on decreasing quickly till epoch 7 where the validation loss became 0. It remained at a constant of 0 till the last epoch showing no loss over the validation data. This means that the model was trained well with train loss of 0.22 at the last epoch and validation loss of 0.00. The higher train loss than validation loss indicates that the model was not overfitted. Rather, a slight misclassification at the train data was present that made the model flexible enough to predict on the unseen validation data.

5.1.2 Error Rate during Training

Error rate represents the actual performance of the model over the validation data. A good classification model must have a high classification rate over the unseen data. In case of evaluating the training of the model, the error rate over the validation data is an important measure of estimating the performance of the deep learning model. The graph in Figure 5.2 shows the trend of error rate over each epoch in the training of the 2 Character ligature set over the validation data.

The error rate was at a high value of 0.55 at first epoch and then it decreased

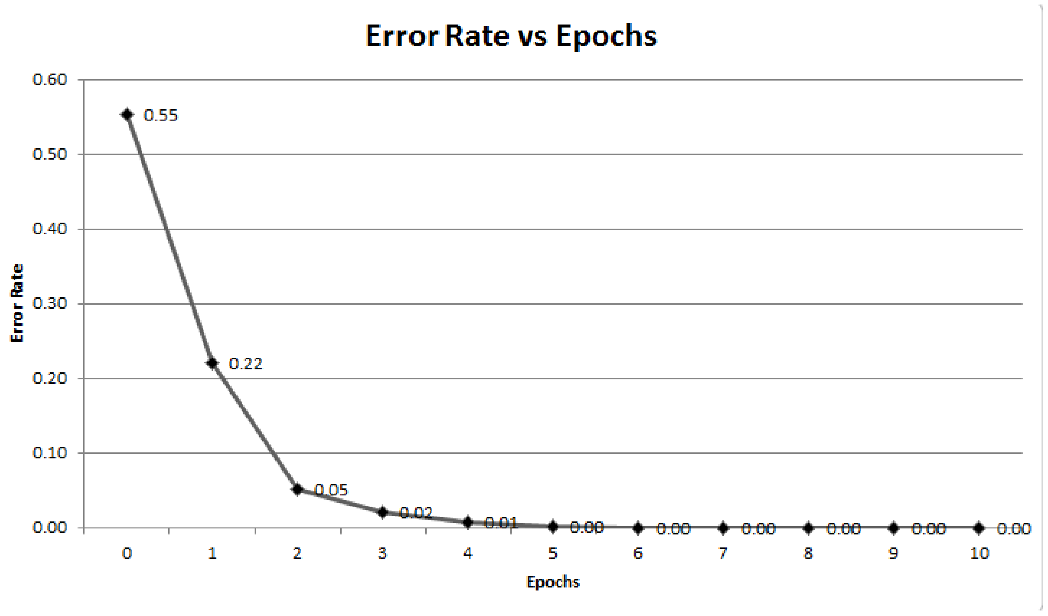


Figure 5.2: Error rate over each epoch (2 Character Ligature Set).

very quickly in the second, third and fourth epoch after which it became 0. A 0 error rate means none of the validation data images were misclassified. The training model had a very high performance over the validation data indicating that it would also predict well over the test data. This indicates that our model performs well in the classification of 2 character length Urdu ligatures.

5.1.3 Training Time

The total training time for training over the 2 character ligature set was 13.98 minutes. The time taken at each epoch is as follows; The training time was not very long for 2 character dataset. But this was because of the use of supercomputer. Similar training was done using COLAB, which took almost 12 hours to complete for the same dataset.

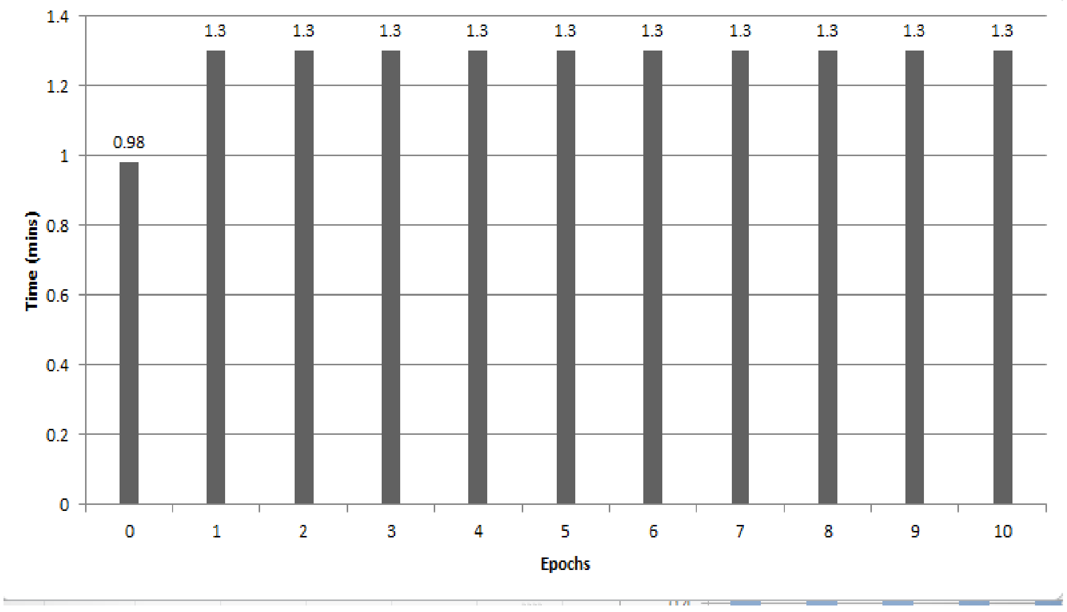


Figure 5.3: Training time with each epoch (2 character ligature set)

5.2 Evaluation of 3 Character Model

Similar Evaluation as done for the 2 character ligature set are performed for the 3 character ligature set. The dataset used was way bigger than the 2 character dataset. The number of ligatures in the 3 character ligature set were;

3 character Ligature Set	**Ligatures**
Train Data	351232
Validation Data	87808

Table 5.2: Ligatures division in 3 character ligature set

The 3 character dataset comprised of 96.55% of the total ligatures in the 2 and 3 character ligatures sets combined. Therefore, training such a large number of ligatures was a difficult task.

5.2.1 Training and Validation Loss

The comparison of training and validation loss is an important measure that tells whether the model was trained well or not. The train and validation loss converges in a high performance deep learning model. But the validation data accuracy should be higher than the train data accuracy.

High validation data accuracy means the model is not overfitted and low train data accuracy means the model is not underfitted. Both the train and validation losses should be on the lower side, but the train loss should not become 0 because in that case the model would become overfitted.

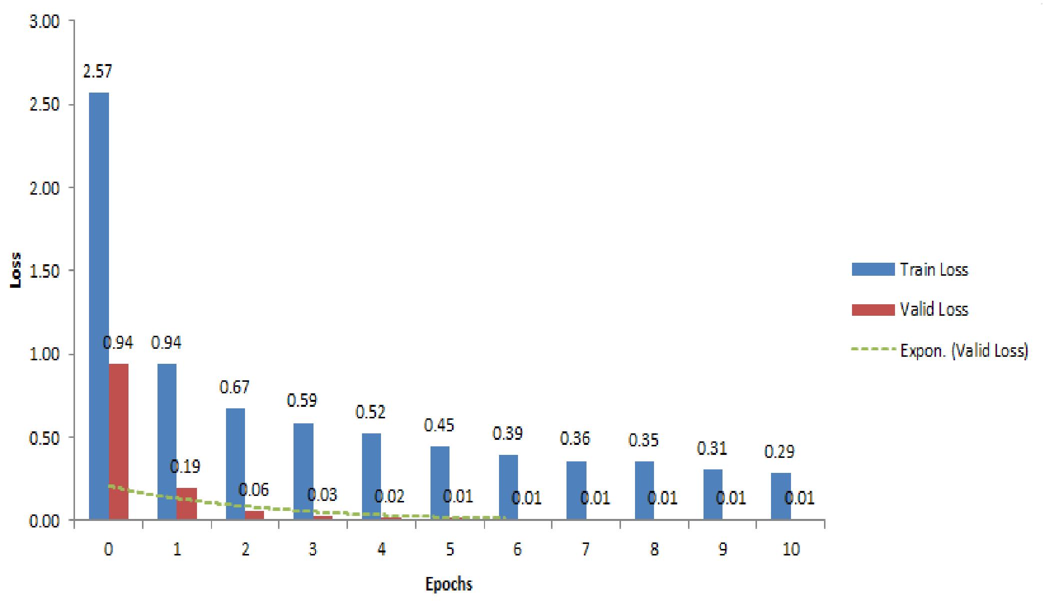


Figure 5.4: Train and Validation Loss at each epoch (3 Character Ligature Set).

The graph in the fig 5.4 shows the validation and train loss at each epoch. The graph clearly depicts that the training model was converging with each epoch. In the first epoch the train and validation losses both were at high value of 2.57 and 0.94 respectively. With each epoch, both the losses decreased drastically. The train loss kept on decreasing till the last epoch and was 0.29 in the last epoch. Whereas the validation loss decreased till epoch 5 and the remained at a constant of 0.01 till the last epoch. Such a small validation loss is a clear indication that the model was trained very well, it was not overfitted and would also produce high test data accuracy.

5.2.2 Error Rate during Training

The trend of error rate with each epoch is shown in Figure 5.5; The trend of error rate to accuracy is completely opposite to each other. The error rate at first epoch was 0.21 and them decreased till epoch 4 and became 0 after

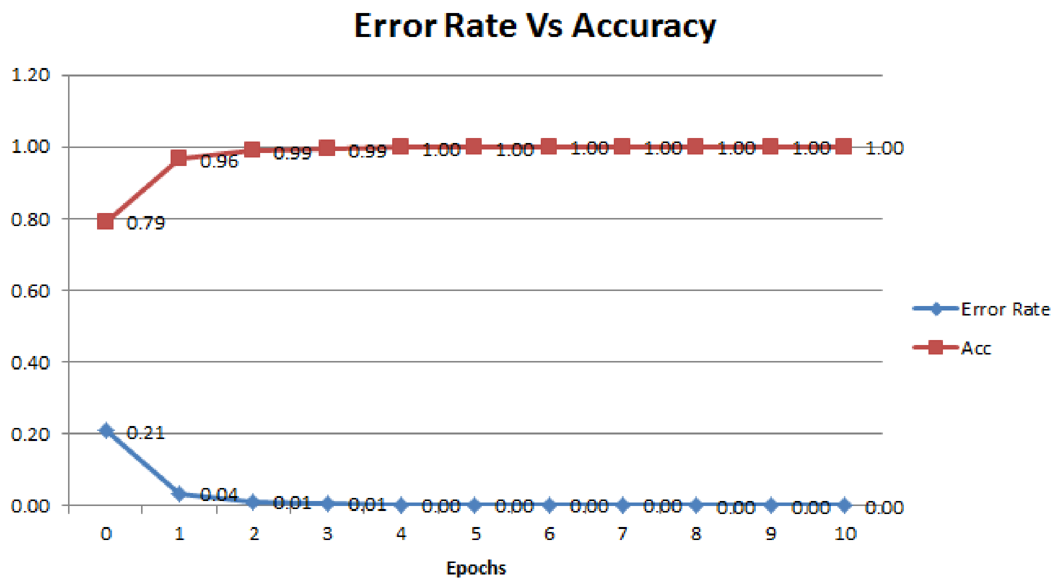


Figure 5.5: Comparison of error rate with accuracy at each epoch (3 Character Ligature set)

that. The reciprocal of error rate was the trend of accuracy. It was low at first epoch which was 0.79 and then increased with each epoch and finally became 1 after which it remained constant. An accuracy of 1 means that the model predicted the ligatures in the validation data with 100% accuracy.

The model showed remarkable results over the validation data and indicates good performance on unseen data i.e., the test data, as indicated in the results section of the thesis.

5.2.3 Training Time

The total training time for 3 character ligature set was approximately 6.4 hours. The time taken by each epoch is shown in the Figure 5.6. The first epoch took 27.36 minutes to complete and later increased in the next epoch. The training time for each epoch was around 35 minutes in each epoch. The time taken indicated the time complexity of our training problem. The large number of ligatures in the form of images required high computational power for processing and recognition. The same task was impossible to be performed over a single local machine. The same training was done using online server COLAB, but it took days of time and still could not be completed. For this reason, the utility of supercomputer was utilized to carry out the task using GPUs which speeded up the classification task tremendously.

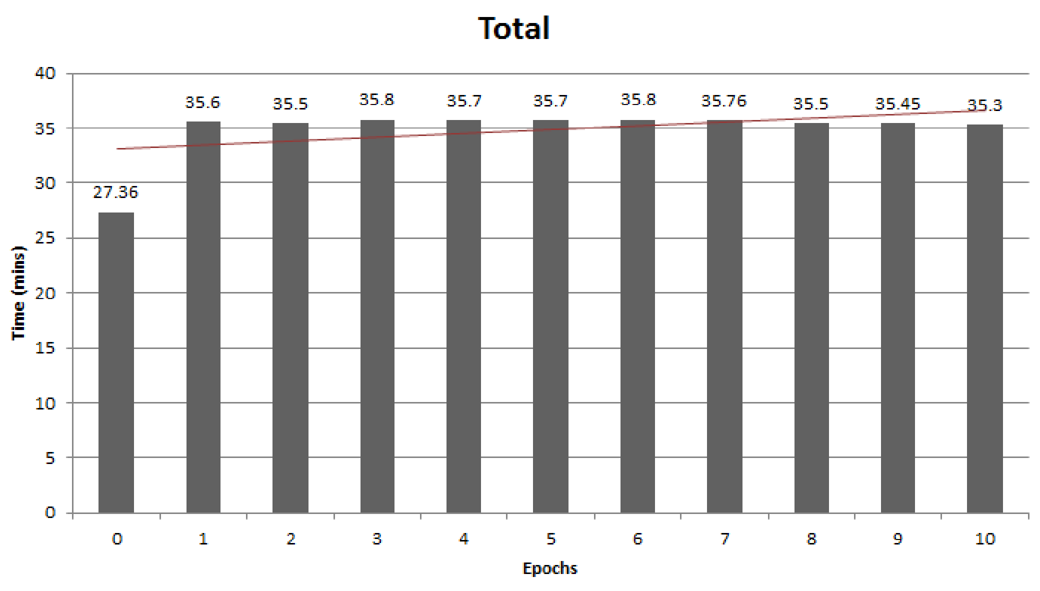


Figure 5.6: Training time with each epoch (3 character ligature set)

5.3 Evaluation of Final Dataset Model

The last model trained was using all the ligatures in two and three characters combined in order to make a classification model capable of classifying multi length ligatures. Though we have limited our research over 2 and 3 character length ligatures, but the method can be easily extended for 4 and more length ligatures. The 'Final Dataset' contained the combination of 2 character ligature set and 3 character ligature set. This made the dataset to contain following number of ligatures.

Combined 2 and 3 Character Ligature Set (Final Dataset)	**Ligatures**
Train Data	363776
Validation Data	90944

Table 5.3: Ligatures division in Final Dataset

5.3.1 Comparison of Final Dataset to 2 and 3 character ligature sets

The final dataset as already mentioned, was a combination of both the 2 and 3 character ligatures for which separate training networks had been trained and validated. The size comparison of these dataset as shown in Figure 5.7 indicates that of all the 2 and 3 character length ligatures, the two character ligature set comprised only 3.45% of the total ligatures, while the 3 character set had 96.55% of the ligatures from the total ligature sets. This means that the 2 character ligature set was 93.1% smaller than the 3 character ligature set. The time and computational complexity to train these networks had a huge difference as can be already seen by the training times of these models.

There is not much difference between the size of 3 character ligature set and final dataset, but the smaller number of 2 character ligatures was a challenge in the final dataset, as the number of ligatures did not form the majority class and could have been misclassified. However, the results have shown the high performance rate even on this dataset, indicating the excellent classification capabilities of the Resnet34 model. The evaluation of these results would be performed in the upcoming sections as well.

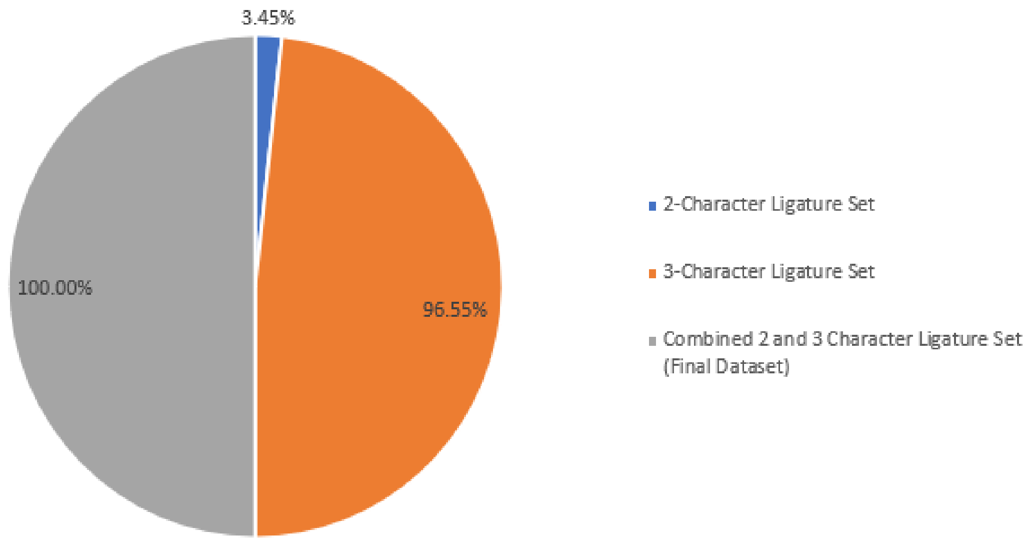


Figure 5.7: Pie Chart representing the size of ligature sets

The Figure 5.8 is a bar chart representation of the distribution of images in each of the ligature sets into train and validation data. The graph clearly depicts that the 2 character ligature set is smaller than the 2 and 3 character ligature sets. There is not much difference between 3 character ligature set and the final dataset.

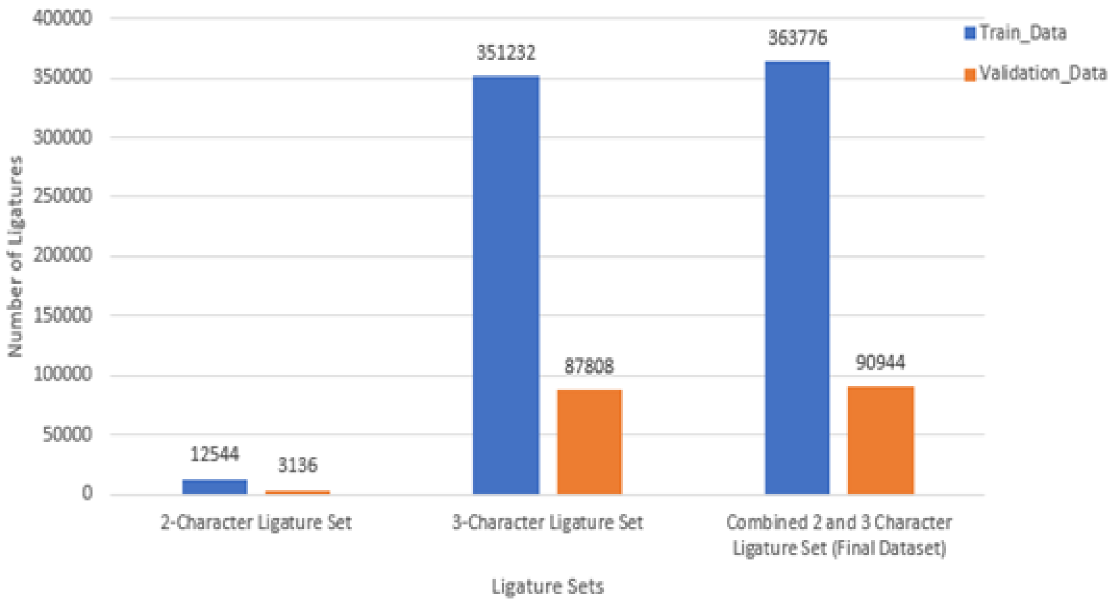


Figure 5.8: Bar chart showing comparison of ligature sets

5.3.2 Training and Validation Loss

The graph that shows the train and validation loss at each epoch for final dataset has the same trend as the training of 2 and 3 character ligatures separately. The train and validation losses decrease with each epoch. The train and validation losses are both high at epoch 0 i.e., 2.71 for train loss and 1.00 for validation. It decreased significantly in the epoch 1 to 1 and 0.25 respectively. The train loss kept decreasing till epoch 9 and increased by only a few decimal places in the last epoch whereas the validation loss became at a constant of 0.01 after epoch 7. The training completed with a train loss of 0.36 and validation loss of 0.01. These results indicate good training model that is not overfitted. The train loss is low but not so much that it could cause underfitting and not too much to cause the model to overfit.

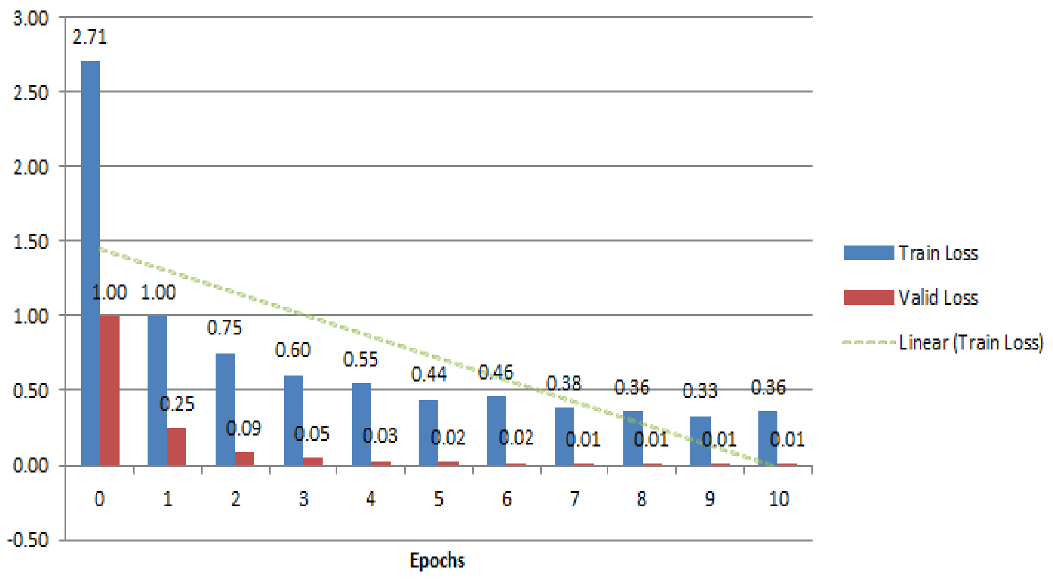


Figure 5.9: Train and Validation Loss at each epoch (Final Dataset).

5.3.3 Error Rate

The model got trained with 0 error rate and 100% accuracy over the validation set. The graph below indicated that the model had an error rate of 0.22 in the beginning but quickly decreased to 0.05 in the very next epoch and became completely 0 over the validation set from epoch 5. It must be noted here that it is not the train loss but the validation error rate. The train loss was 0.36, whereas the error rate over the dataset turned out to be 0 after training, indicated that the slight train loss made the model flexible to predict on dataset on which it had not been trained.

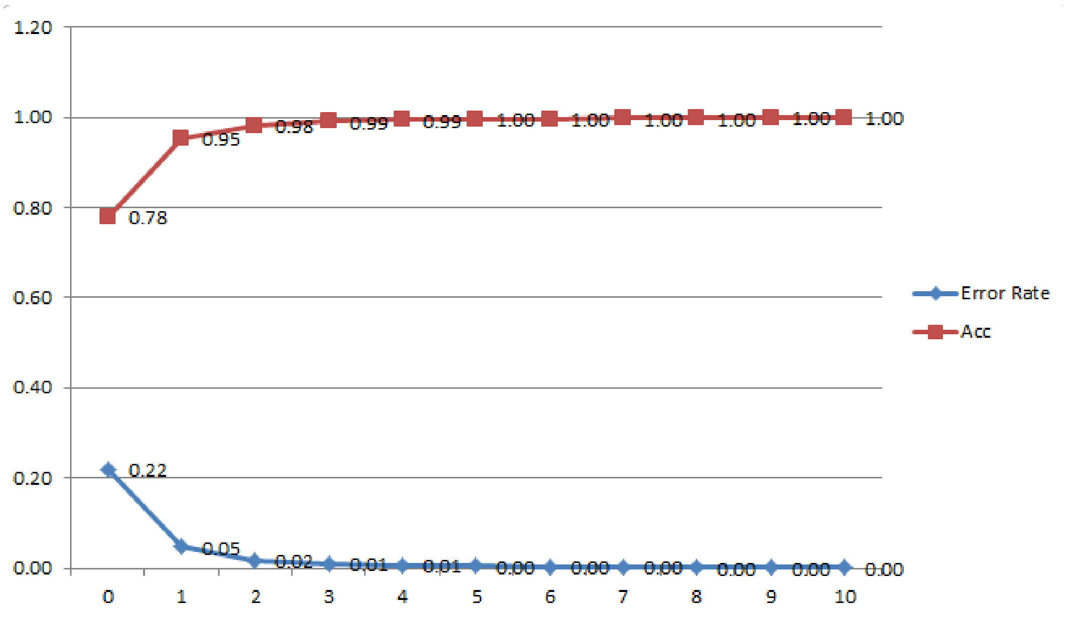


Figure 5.10: Comparison of error rate with accuracy at each epoch (Final Dataset)

5.3.4 Training Time

The total training time was 6.6 hours approximately with each epoch taking an average of 37 minutes. The training time for final dataset was almost similar to the 3 character ligature set. This accounts to the fact that 96.5% of the final dataset contained the 3 character ligature images. Yet, the presence of 2 character ligatures cannot be ignored but the small number of images in the 2 character ligature set did not affect the training time very drastically.

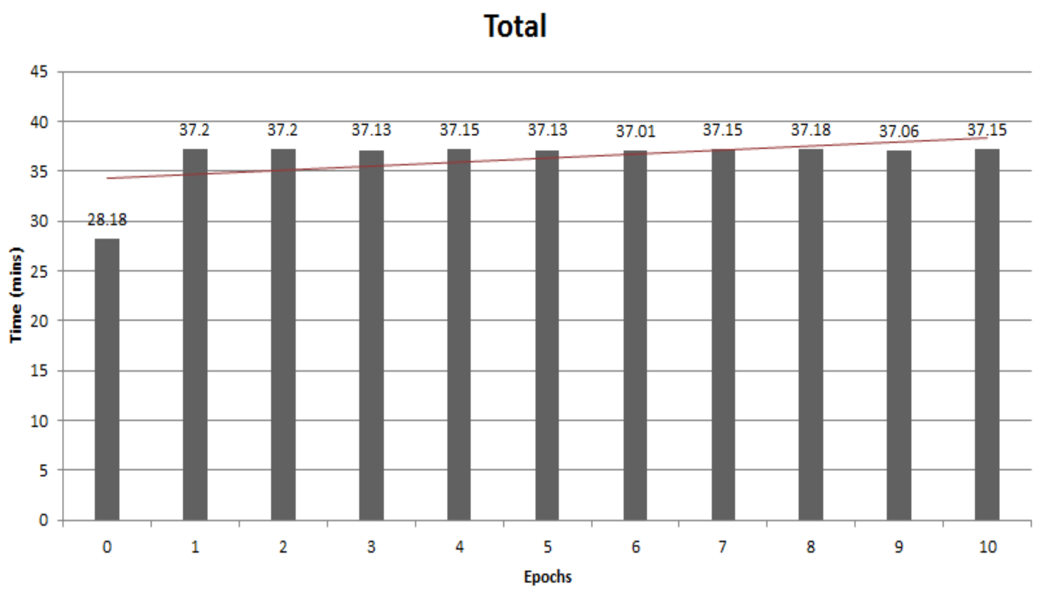


Figure 5.11: Training time with each epoch (Final Dataset)

5.4 Evaluation on Test Data

The above mentioned sections evaluated the results obtained during the training process of the model. The real evaluation which proves that the model is performing good or not depends upon its results over the test data which is completely unseen to the model and had not been used for training.

5.4.1 Evaluation Metrics

The evaluation metrics used were;

The first step involved before evaluating the results is to create a confusion matrix of the result. A confusion matrix compares the results of the target data to the predicted data [56].

	Actual Positive Class	Predicted Negative Class
Actual Positive class	TP (True Positive)	FN (False Negative)
Predicted Negative Class	FP (False Positive)	TN (True Negative)

Table 5.4: Confusion Matrix

The metrics calculated from this confusion matrix are:

Accuracy: Accuracy calculates a ratio among the number of images correctly classified by the total number of instances in the test set [56].

Accuracy (acc) = TP+TN / TP+TN+FP+FN

Error Rate: It is the ratio of the misclassifications [56].

Error Rate (ER) = FP+FN/ TP+TN+FP+FN

Precision: It is a measure that gives the ratio of the correctly predicted positive class to the total positive classes predicted by the model [56].

Precision (P) = TP/ TP+FP

It is a better measure than accuracy because a model will have a higher accuracy if no negative class was predicted in an unbalanced dataset where negative instances were very few.

Recall:

It gives the probability of the positive instances correctly classified from the total number of correctly classified instances [56].

Recall (R) = TP /TP+TN

F1-Score: It is the harmonic mean between recall and precision. The metrics of recall and precision are biased. Precision is affected by the number of false positives, if the FP is high, precision would be low. Similarly, Recall is affected by TN, Recall is high is the number of TN is low. Inorder to remove this effect and averaged measure known as F1 is used, which is not affected by any of the measures [56].

F1- Score (F1) = 2 * P * R / P+R

The above mentioned metrics are for binary class classification problems, but these can also be used for multi-class classification problem. In a multi-class classification problem one vs all approach is used, where each class is considered as positive and rest as negative. The metrics mentioned above are calculated for each class and are then averaged to provide an overall evaluation of the model [56].

We evaluated all our three trained models which were 2, 3 character ligature sets and final dataset over test sets. The outcome of testing and their explanation is discussed in the following sections.

5.4.2 Evaluation of 2 Character Ligature Set

The test data for 2 character ligature set contained 3138 images generated as mentioned in the methodology section of this thesis. The images in this dataset were used for prediction over the trained Resnet34 model. The predictions were stored and compared to the target data to evaluate the model.

Accuracy: The model's accuracy was 0.96 which means that 96% of the images were correctly recognized by the system. The model showed a remarkable accuracy of 96% showing how well it could predict on unseen data.

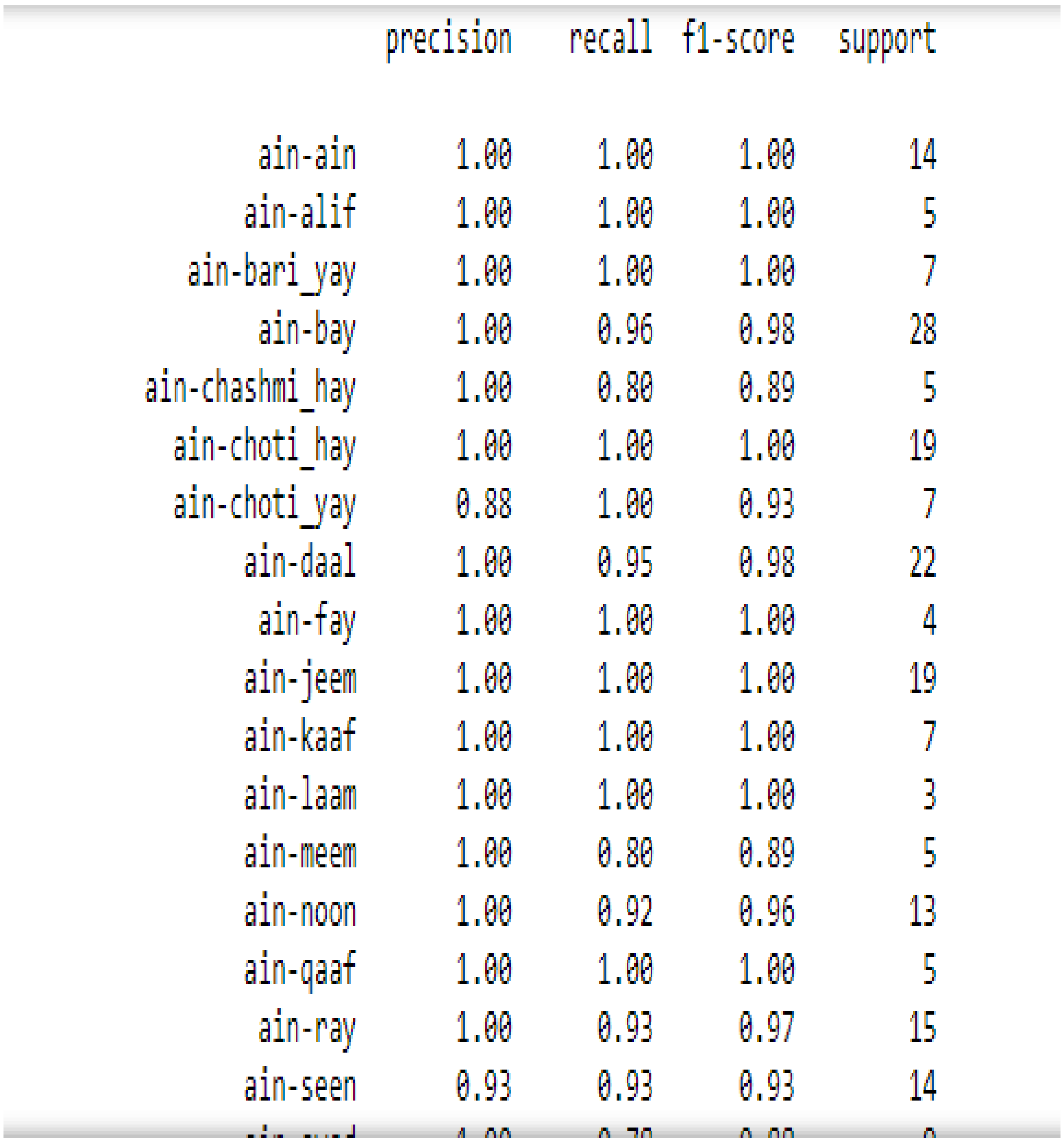
Error Rate: The model showed and error rate of 0.4 meaning that out of the total only 40% of the images were not correctly recognized.

Macro- Average Precision: The model's average precision was 0.91. It means that on the average 90.1% of the classes were correctly predicted from the total positives predicted.

Macro- Average Recall: The average recall was 0.9 showing that 90% of the ligatures were correctly recognized out of the total recognized images.

F1-Score: The F1-score was 0.90 indicating 90% of the images correctly classified without the bias of recall or precision. This high value indicated that both the false positives and false negatives were low.

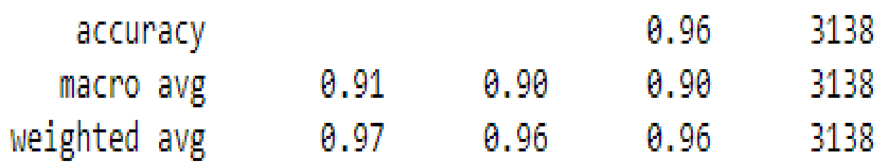
The output of the classification report gives the evaluation metrics of the model. For every class the precision, recall and F1 score is calculated. The fig shows few classes and their evaluation metrics.



```
                 precision    recall  f1-score   support

         ain-ain      1.00      1.00      1.00        14
        ain-alif      1.00      1.00      1.00         5
    ain-bari_yay      1.00      1.00      1.00         7
         ain-bay      1.00      0.96      0.98        28
 ain-chashmi_hay      1.00      0.80      0.89         5
   ain-choti_hay      1.00      1.00      1.00        19
   ain-choti_yay      0.88      1.00      0.93         7
        ain-daal      1.00      0.95      0.98        22
         ain-fay      1.00      1.00      1.00         4
        ain-jeem      1.00      1.00      1.00        19
        ain-kaaf      1.00      1.00      1.00         7
        ain-laam      1.00      1.00      1.00         3
        ain-meem      1.00      0.80      0.89         5
        ain-noon      1.00      0.92      0.96        13
        ain-qaaf      1.00      1.00      1.00         5
         ain-ray      1.00      0.93      0.97        15
        ain-seen      0.93      0.93      0.93        14
```

Figure 5.12: Classification Report of 2 character ligature set Testing

The averaged recall and precision over the test data for 2 character was as follows:



```
    accuracy                           0.96      3138
   macro avg       0.91      0.90      0.90      3138
weighted avg       0.97      0.96      0.96      3138
```

Figure 5.13: Averaged metrics for evaluation (2 Character Ligature Set)

The graph below gives a better visual understanding of the above shown metrics;

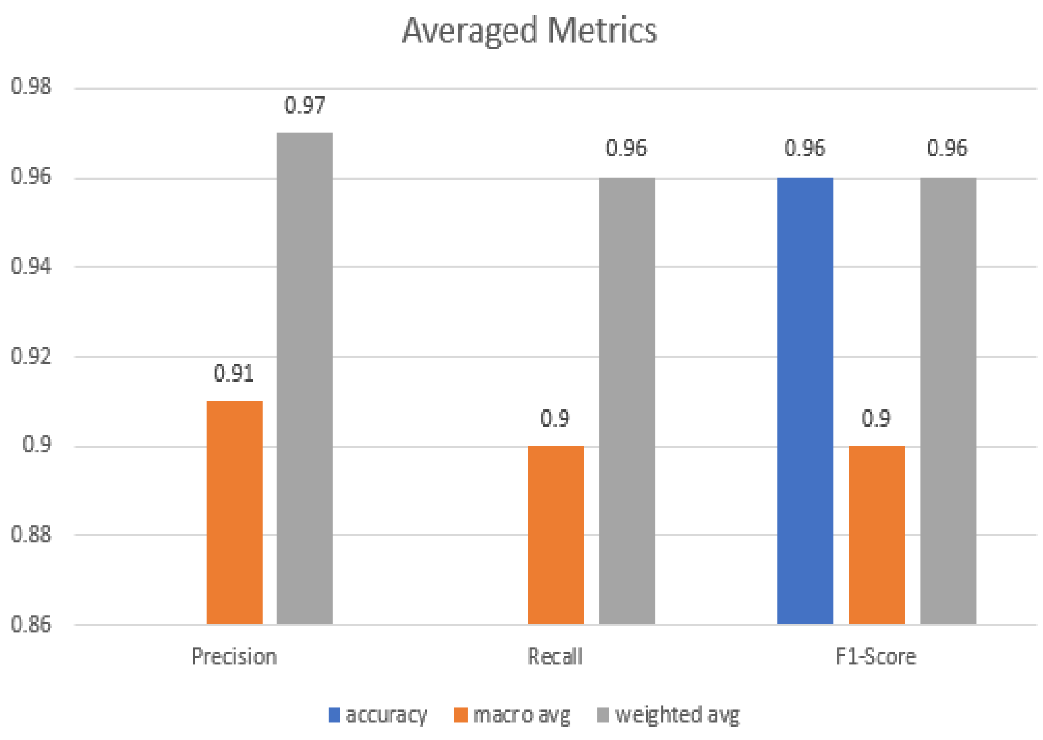


Figure 5.14: Metrics (2 Character Ligature Set)

All the metrics show that the trained model performed well over the unseen data with 0.96 accuracy and 0.91, 0.9 precision and recall respectively.

A scatter plot representing the outcomes of prediction showing both the target and the predicted values over the 2 character test set is shown;

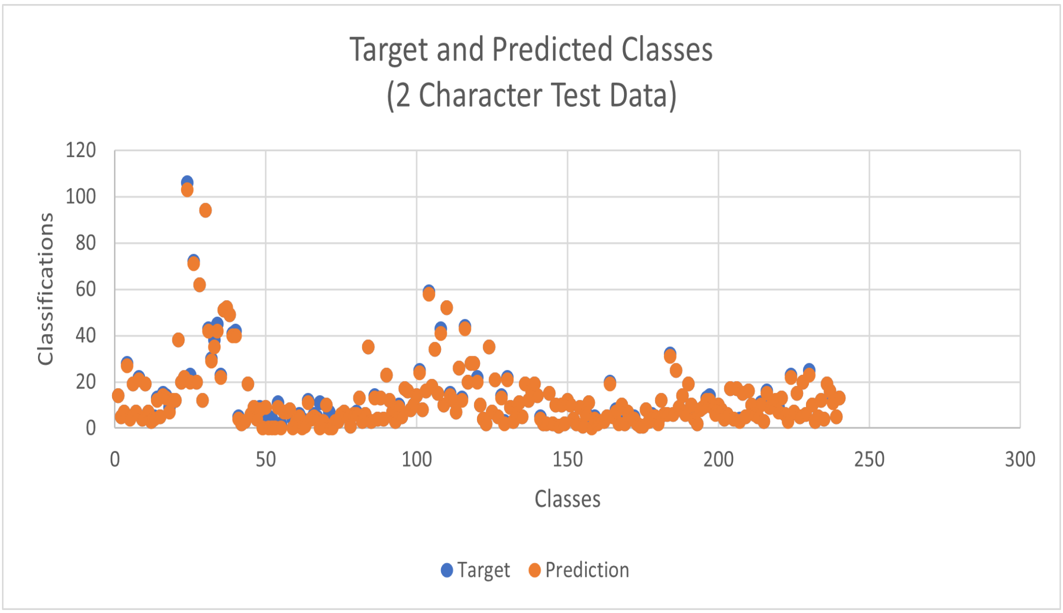


Figure 5.15: Scatter plot of Target and Predicted Classes (2 Character Ligature Set)

This plot shows a correlation between the classes in the test data and the predictions on each of the class. In the 2 character ligature test set a total of 3138 images from 240 classes were present. For all 240 classes, the predictions were made using the trained model and these predictions in the form of sum of images correctly classified are represented here as a mark. It is a very self-explanatory graph that clearly shows that most of the predicted classes (represented by orange mark) overlap the target indicating that the predictions were correct. Few blue labels indicate the mis-classifications.
The correlation between the target and the predicted labels has been generated as shown in figure 5.14.
A linear curve is present between the two. The class labels when compared to the target class show that all the target and the predicted labels were same meaning they were predicted accurately. This straight line is a clear indication of correctly labeled 2 ligatures from the test set.

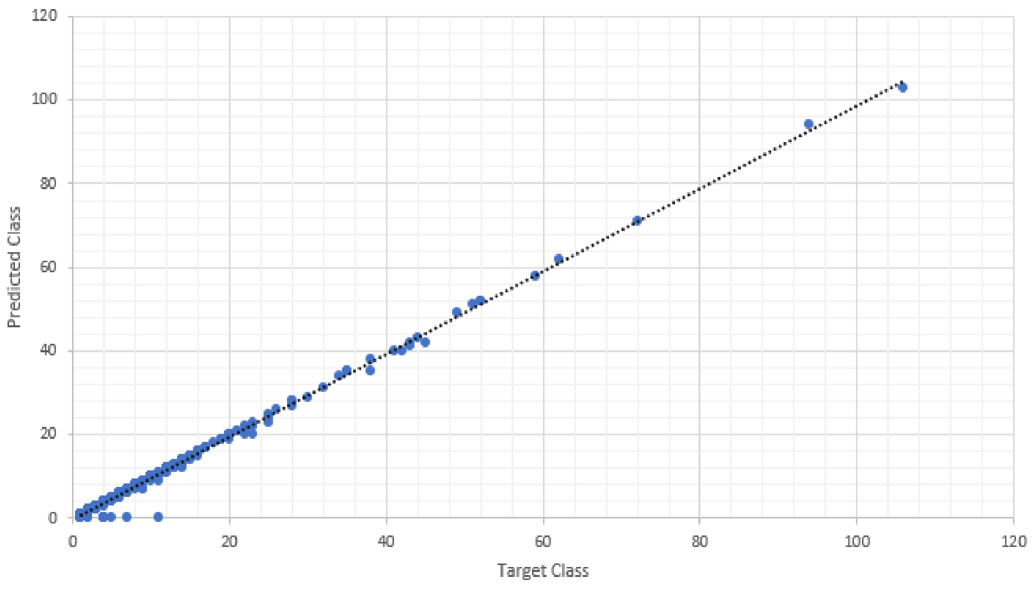


Figure 5.16: Scatter Plot (Target vs Predicted Classes) – 2 Character Ligature Set

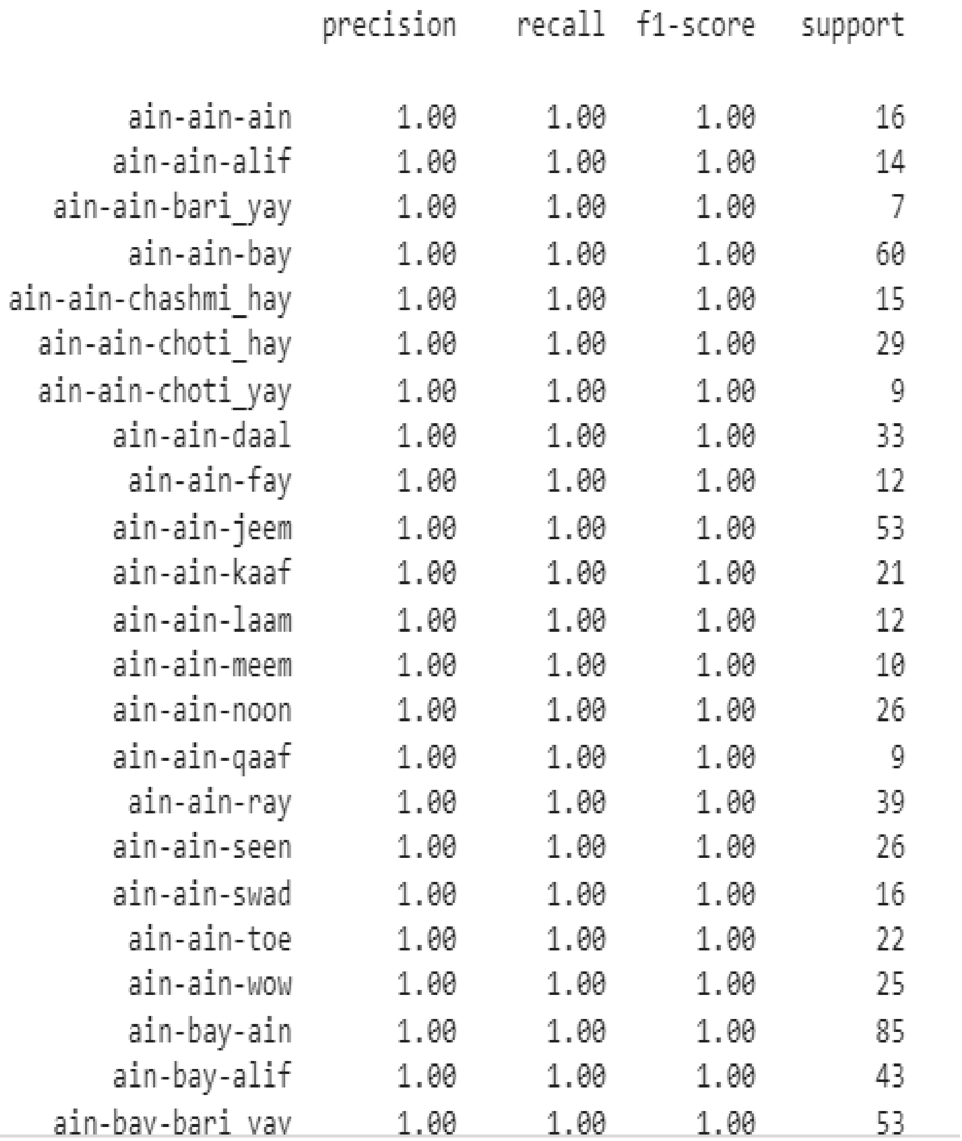
5.4.3 Evaluation of 3 Character Ligature Set

The 3 character ligature set contained 87807 ligature images in the test set. The size of the test set was 20% to the train set and equal to the size of the validation set. The prediction resulted in the classification of the unseen ligatures with a 99.70% accuracy.

```
Accuracy of the classifier is: 99.70%
```

Figure 5.17: Accuracy score output

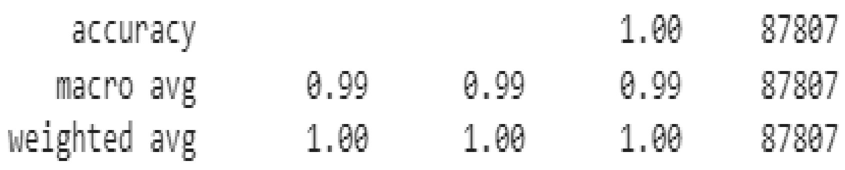
The method to calculate the precision, recall and F1 score was the same as done for 2 character ligature set. For each class the metrics were calculated as shown below;



```
                     precision    recall  f1-score   support

        ain-ain-ain       1.00      1.00      1.00        16
       ain-ain-alif       1.00      1.00      1.00        14
   ain-ain-bari_yay       1.00      1.00      1.00         7
        ain-ain-bay       1.00      1.00      1.00        60
ain-ain-chashmi_hay       1.00      1.00      1.00        15
  ain-ain-choti_hay       1.00      1.00      1.00        29
  ain-ain-choti_yay       1.00      1.00      1.00         9
       ain-ain-daal       1.00      1.00      1.00        33
        ain-ain-fay       1.00      1.00      1.00        12
       ain-ain-jeem       1.00      1.00      1.00        53
       ain-ain-kaaf       1.00      1.00      1.00        21
       ain-ain-laam       1.00      1.00      1.00        12
       ain-ain-meem       1.00      1.00      1.00        10
       ain-ain-noon       1.00      1.00      1.00        26
       ain-ain-qaaf       1.00      1.00      1.00         9
        ain-ain-ray       1.00      1.00      1.00        39
       ain-ain-seen       1.00      1.00      1.00        26
       ain-ain-swad       1.00      1.00      1.00        16
        ain-ain-toe       1.00      1.00      1.00        22
        ain-ain-wow       1.00      1.00      1.00        25
        ain-bay-ain       1.00      1.00      1.00        85
       ain-bay-alif       1.00      1.00      1.00        43
   ain-bay-bari_yay       1.00      1.00      1.00        53
```

Figure 5.18: Classification Report of 3 character ligature set Testing

The results of classification over the test set for 3 character ligature set gave a remarkable accuracy of 99.7% approximately equal to 1. This result directly shows how well the training model can predict on the unseen data.



```
    accuracy                          1.00     87807
   macro avg       0.99      0.99      0.99     87807
weighted avg       1.00      1.00      1.00     87807
```

Figure 5.19: Averaged metrics for evaluation (3 Character Ligature Set)

All the evaluation measures of precision, recall and F1-score were 0.99 showing that the trained model performed brilliantly well over the test set with very few misclassifications. The bar graph shows these evaluation metrics.

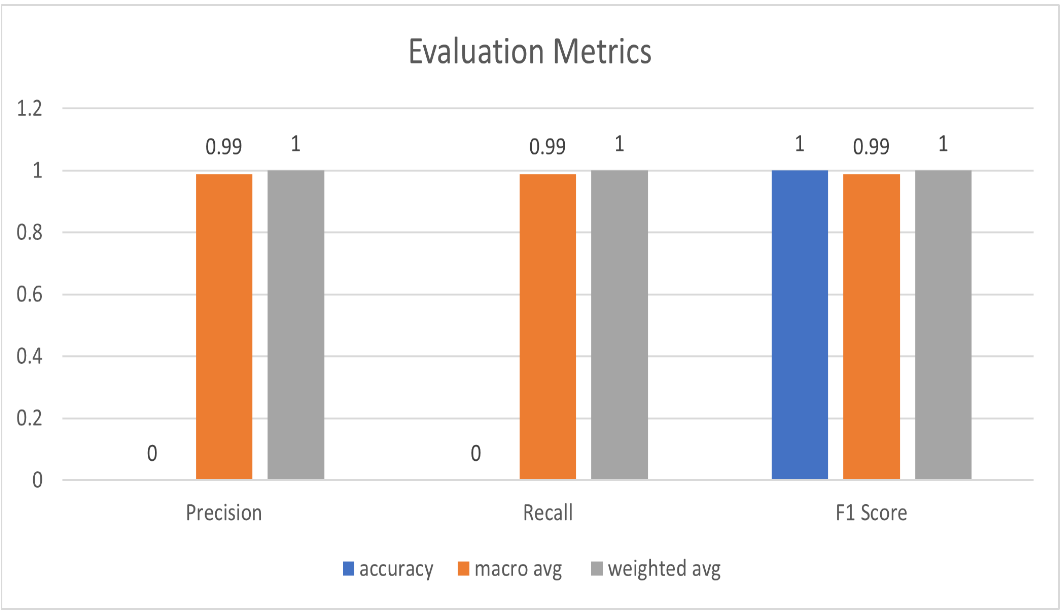


Figure 5.20: Metrics (3 Character Ligature Set)

Visual Explanation of Test Data Classification

The major requirement in a classification evaluation is to determine how well the trained model predicted the target classes. If there is difference between the class label of the target and predicted class, it means that there was misclassification, and the model has not performed well. The aim of this section is to represent the results of predictions on the 3 character ligature test set. There were 87807 images belonging to 2875 classes. Out of the 87807 images, 87543 were correctly classified and only 264 images were misclassified. The results are shown in the form of a graph to show a concise explanation of the results.

Graph in Figure below shows the target and the predicted classes plotted. The graph can be seen covered with ‘orange’ marks of the predicted class while the target classes in ‘blue’ can hardly be seen. This overlap shows that the target and the predictions were almost exactly the same with only few blue marks that can be seen at very close examination of the graph. The ‘red’ circles show few of the sparse misclassifications.

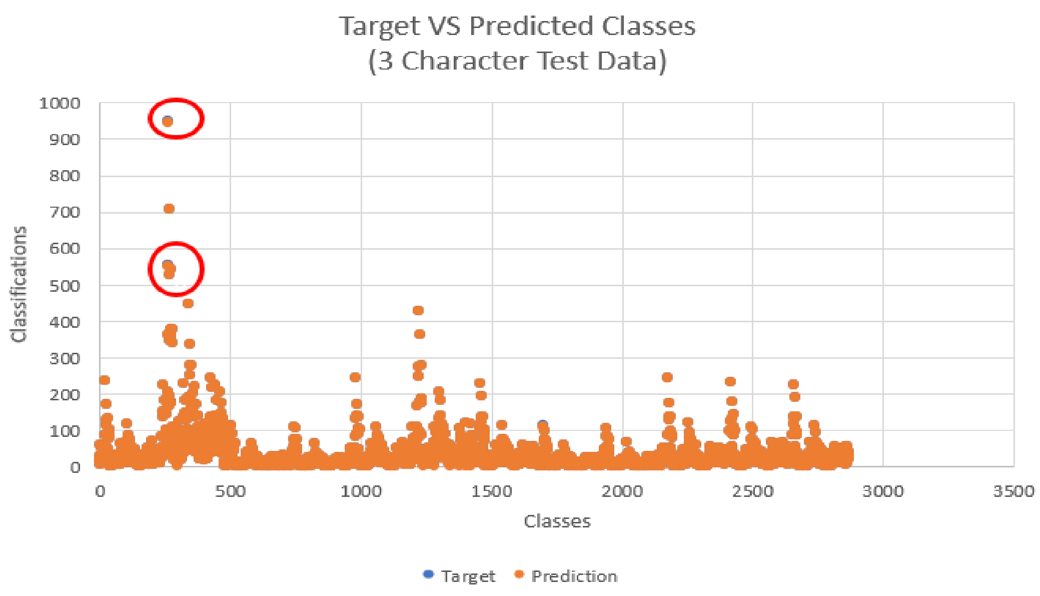


Figure 5.21: Scatter plot of Target and Predicted Classes (3 Character Ligature Set)

The correlation between the target and the predicted classes is depicted in Figure 5.21. The number of images in the target and the predicted classes are taken on the x and y axis respectively. The graph shows a linear trend again showing that the target and the predicted labels were same and therefore were plotted on the same point on the graph. The linearity of the graph is positive with number of images in the target and prediction for each class to be same. If the target and the predicted label would have been different then this data points would have been arbitrary depicting variations between target and the predicted labels.

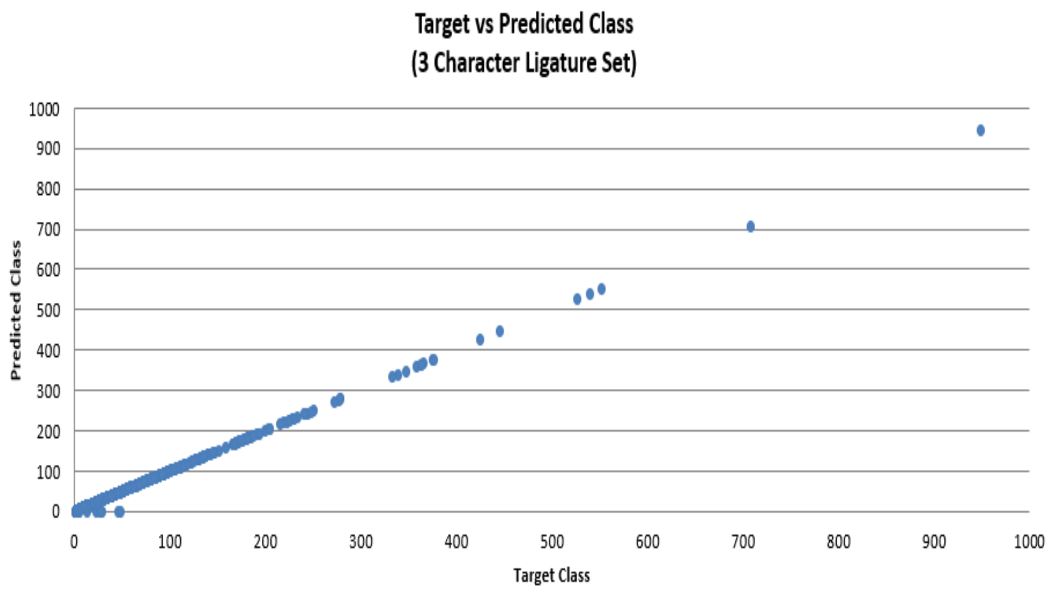


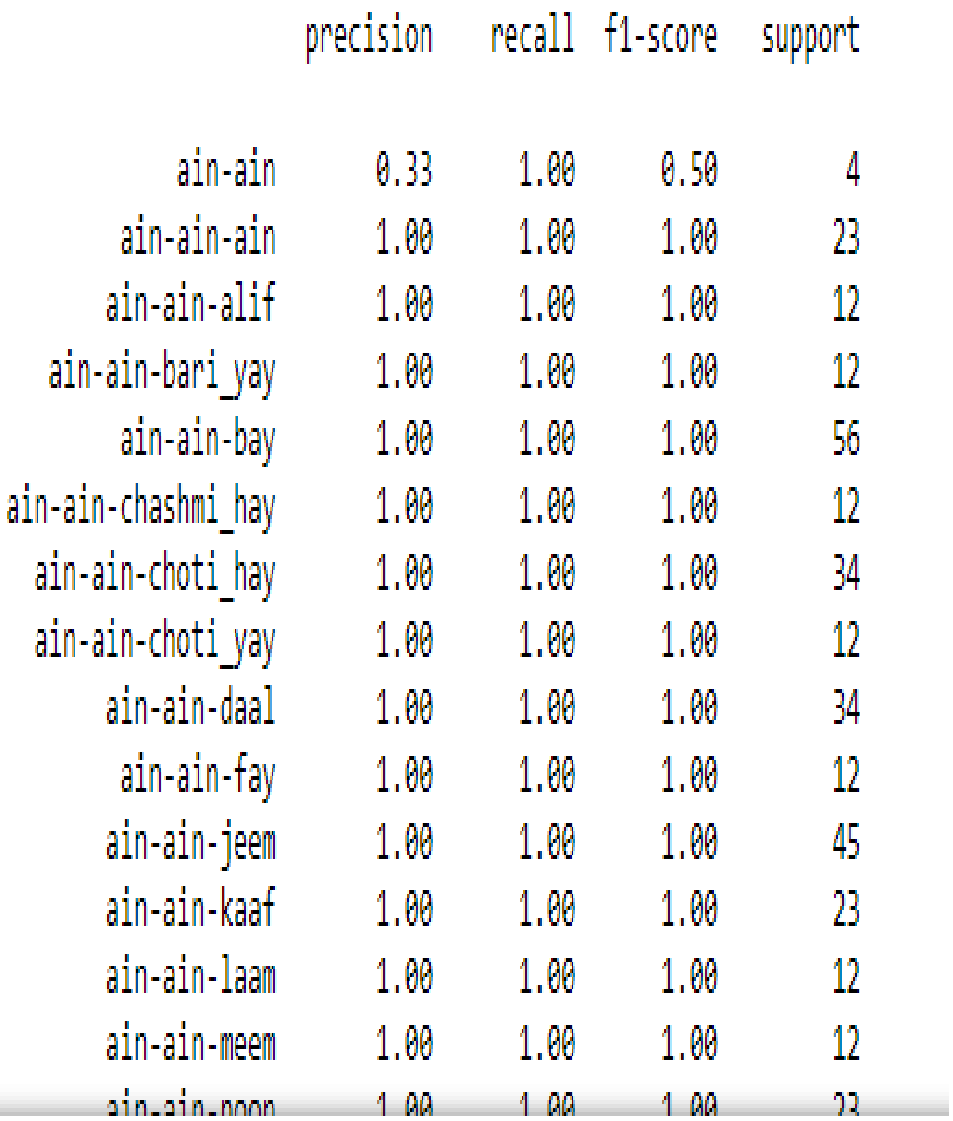
Figure 5.22: Scatter Plot (Target vs Predicted Classes) – 3 Character Ligature Set

5.4.4 Evaluation of Final Dataset

The final dataset contained ligatures of both 2 and 3 character ligature set. The evaluation of the trained model over this dataset was our main objective. To create a model that correctly recognize both the ligatures of 2 and 3 character ligature correctly was our classification task. After successfully training the Resnet34 model over the final dataset, now comes the point to evaluate it over the test set.

The test set for evaluation of the Final Dataset model contained 92474 images from 3976 classes. Out of the total images in the test set 89838 images were correctly classified and only 2636 images misclassified comprising only 2% of the images.

The precision, recall and F1-score was calculated foe each class, as it is a multi-class classification problem. These measures were then averaged to generate a single value for these evaluation metrics. Few instances of the classification results from the classification report, along with the evaluation measures are shown in the Figure 5.23 and 5.24.



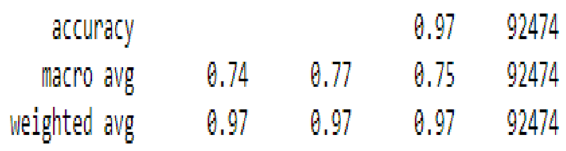
```
                    precision    recall  f1-score   support

          ain-ain       0.33      1.00      0.50         4
      ain-ain-ain       1.00      1.00      1.00        23
     ain-ain-alif       1.00      1.00      1.00        12
 ain-ain-bari_yay       1.00      1.00      1.00        12
      ain-ain-bay       1.00      1.00      1.00        56
ain-ain-chashmi_hay     1.00      1.00      1.00        12
 ain-ain-choti_hay      1.00      1.00      1.00        34
 ain-ain-choti_yay      1.00      1.00      1.00        12
     ain-ain-daal       1.00      1.00      1.00        34
      ain-ain-fay       1.00      1.00      1.00        12
     ain-ain-jeem       1.00      1.00      1.00        45
     ain-ain-kaaf       1.00      1.00      1.00        23
     ain-ain-laam       1.00      1.00      1.00        12
     ain-ain-meem       1.00      1.00      1.00        12
     ain-ain-noon       1.00      1.00      1.00        23
```

Figure 5.23: Classification Report of Final Dataset Testing

The model gave an accuracy of 0.97 which means that 97% of the instances were correctly classified out of the total instances classified.

The averaged precision and recall were 0.74 and 0.77 respectively but the weighted averages show that average precision and recall of the model was 0.97.

Weighted precision in the classification task of ‘Final Dataset; was more important than average precision because all the classes did not have the equal number of samples in the test data. The test data contained images from both 2 and 3 character ligatures with only 20% of the representation in terms of size from the



```
    accuracy                           0.97     92474
   macro avg       0.74      0.77      0.75     92474
weighted avg       0.97      0.97      0.97     92474
```

Figure 5.24: Averaged metrics for evaluation (Final Dataset)

training data.

Therefore, it was important to give more weightage to the majority classes based on the classes in the train set. Hence, the prediction over the test data shows that the model had a precision of 0.97 meaning that 97% of the images were correctly classified out of the total positively classified images. The recall was also 0.97 which shows that 97% of the images were correctly classified as positive class out of the total positive and negative classes correctly classified.

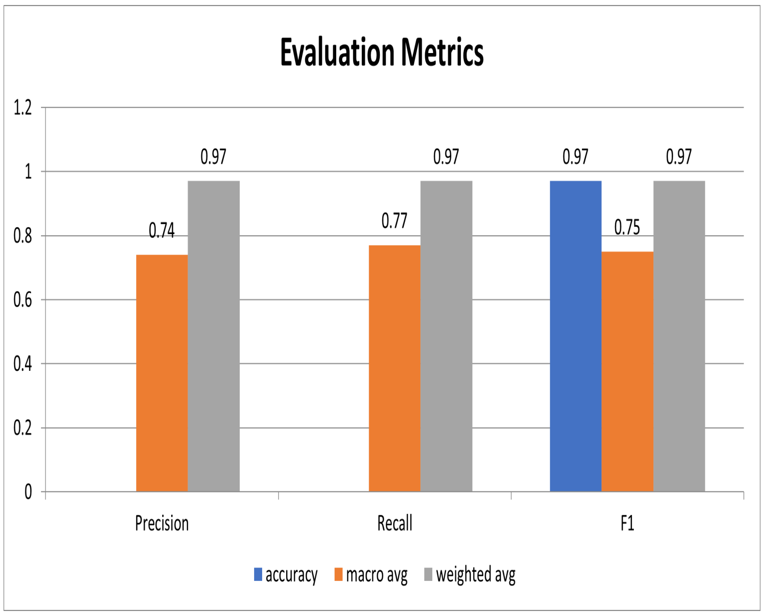


Figure 5.25: Metrics (Final Dataset)

Visual Explanation of Test Data Classification:
The representation of classifications over each class among the 3976 classes in the final dataset using tabular format would not explain the outcomes as better as a graph. The graph in fig 5.26 plots the target and the predicted classes.

The overlap where target class could not be seen shows the correct classifications were the presence of target data point on the graph show that those target points were not classified correctly.

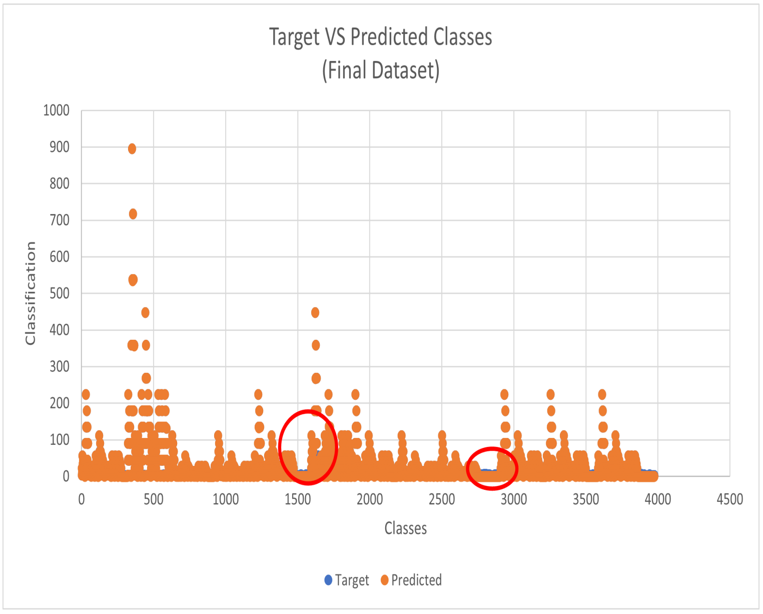


Figure 5.26: Scatter plot of Target and Predicted Classes (Final Dataset)

The scatter plot for evaluation of Final dataset depicts the same trend as the 3 character ligature model prediction. Again, the predicted and the target labels are highly overlapping and misclassifications can be scarcely seen. The overlap is an indication of correct prediction that's why the target and the predicted image lie at the same position in the graph. The target labels can be rarely seen as indicated by the 'red' circles. The points show slight 'blue' datapoints showing target data underneath the predicted datapoints. The rare points highlight the few misclassifications. Such, a high overlap between the target and the predicted values indicates that the error on the test set was extremely low, proving the higher classification accuracy of the trained model.

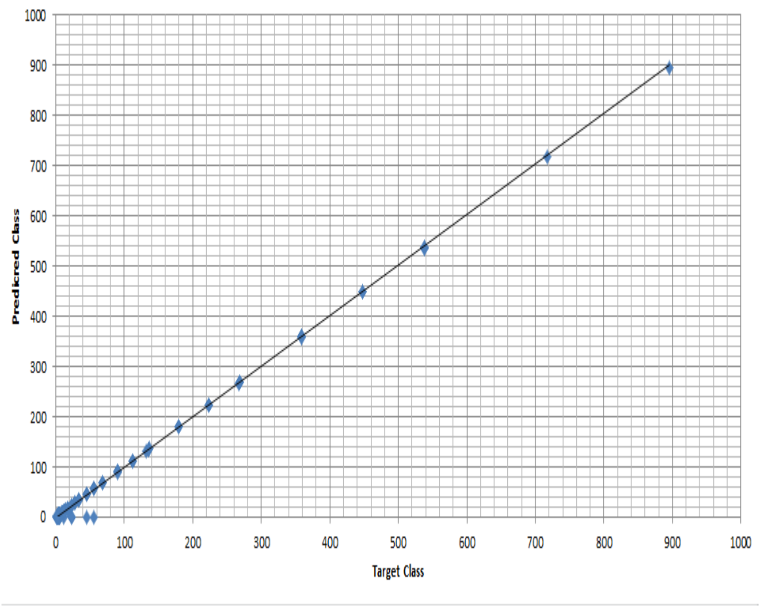


Figure 5.27: Scatter Plot (Target vs Predicted Classes) – Final Dataset

The above graph is among the target and the predicted class. Each mark on the graph is an indication of the prediction. For every class, the total number of correctly classified images are taken on the y axis and the total number of target images for that class are present on the y-axis. The target and the predictions have a positive linearity. The datapoints lie on a straight line which shows that the number of correctly classified images increases in the same trend as the corresponding images in the ground truth. The callouts at each datapoint indicate that almost all the values for each datapoint are same on the x and y axis showing that the number of images correctly classified for a single class are same as the number of images in the target class. Not only the number but this data has been generated based on the class labels and their target and predicted data. Therefore, this linearity is a proof of the correct predictions of our network.

5.5 Discussion

The above mentioned sections explained the results of evaluation separately on the ligature sets. This section compares the results of evaluations of the three ligature sets and discusses in detail the reasons for these results and how the proposed method performs well than the existing methods.

We had three ligature sets and we performed training of three separate networks. In order to compare the performances of these three networks and to identify the network which performed well among them.

5.5.1 Comparison based on correct predictions and misclassifications:

Among the three datasets, 2 character ligature test set was the smallest, it contained 3138 images out of which 3020 were correctly classified and 118 were misclassified giving and accuracy of 96%. 3 Character ligature set was next bigger in size with 87807 ligature images out of which 87543 were correctly recognized by the network and 264 were misclassified with a remarkable accuracy of 99%.

The largest test set was of the Final dataset with 92747 images among which the model correctly classified 89838 images and 2636 images were misclassified giving an accuracy of 97%.

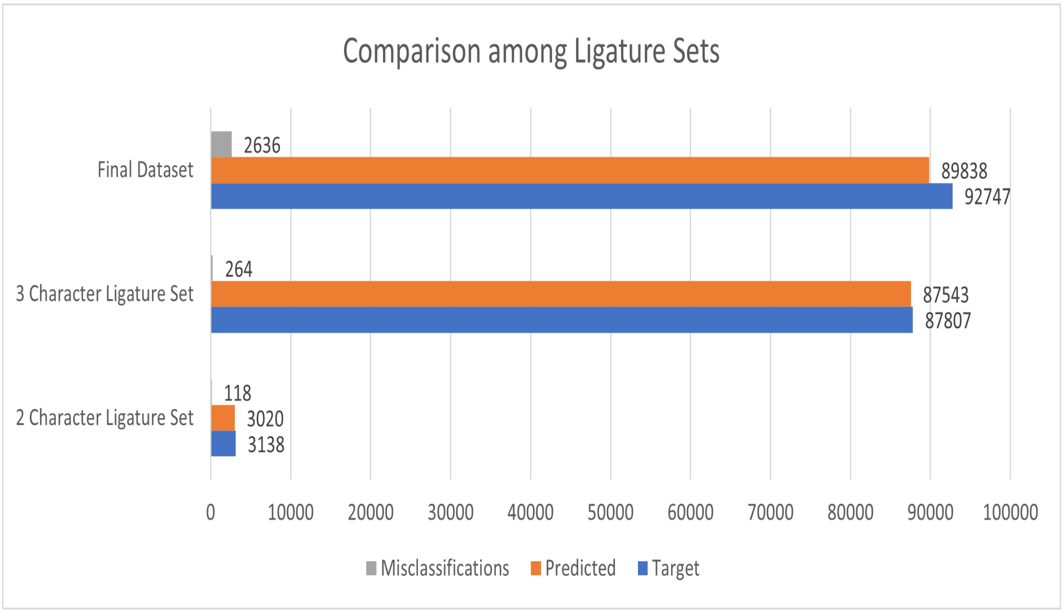


Figure 5.28: Bar chart showing comparison among all the ligature sets

5.5.2 Comparison of Accuracy

The final number which summarizes the overall performance of a deep learning model is its accuracy. The model accuracies of the 3 networks trained for 2 character ligature set, 3 character ligature set and the final dataset is shown in the following figure;

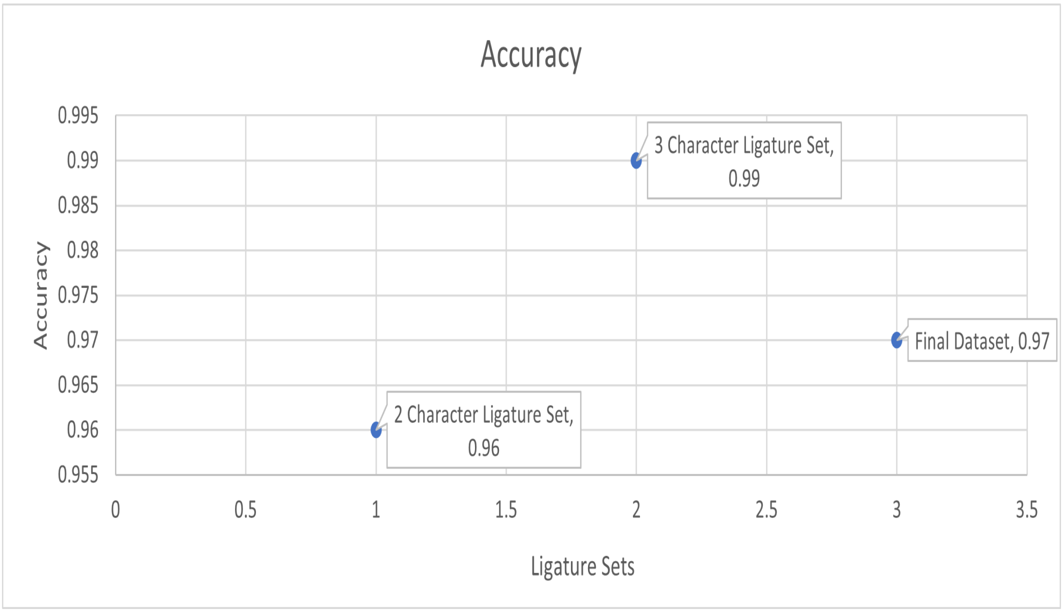


Figure 5.29: Comparison of accuracies of the ligature sets

The overall accuracies of all the three networks were very high. All the networks had accuracies above 95%. Showing that the models performed well over the test sets. A deeper look into their accuracies show that among all the ligature sets, the 3 character ligature set had the highest accuracy of 99%. The second highest accuracy was of the Final dataset which was 97% and the 2 Character ligature set gave an accuracy of 96% which though is very high but was not very high when compared to the other 2 training models.

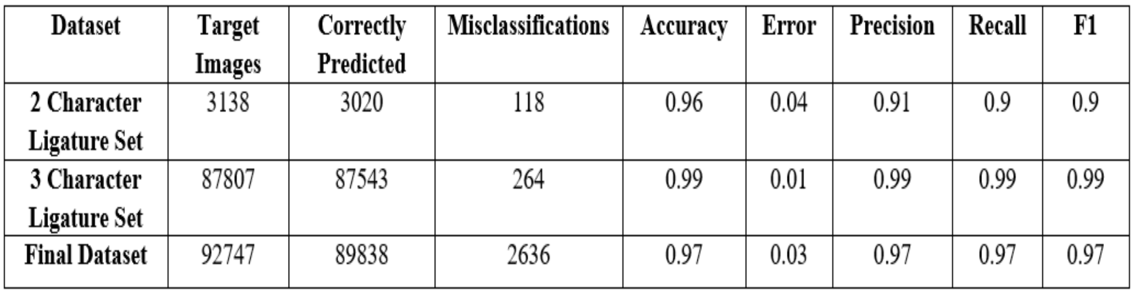
This variation in the accuracies was due to the size variations of these datasets. The 2 character dataset was very small as compared to the other datasets. For a deep learning model, the number of images need to be high. The 2 character ligature set was trained on small number of images and therefore had the lowest accuracy among the other 2 ligature sets.

2 character ligature set had the highest accuracy even though the number of images in the 3 character train set were less as compared to the final dataset train set. The reason being that the training images for 3 character dataset were purely composed of three ligatures, there were not many complexities as in the Final

Dataset. The Final Dataset had a lot of structural variations in terms of the images and the number of classes were high for the final dataset. This increased structural and class complexity accounted for higher accuracy of the three character ligature set than the final dataset which contained both the ligatures from 2 and 3 character ligature sets.

5.5.3 Summary of the results

The table below, summarizes the results of testing all the three ligature sets;



Dataset	Target Images	Correctly Predicted	Misclassifications	Accuracy	Error	Precision	Recall	F1
2 Character Ligature Set	3138	3020	118	0.96	0.04	0.91	0.9	0.9
3 Character Ligature Set	87807	87543	264	0.99	0.01	0.99	0.99	0.99
Final Dataset	92747	89838	2636	0.97	0.03	0.97	0.97	0.97

Table 5.5: Summary of test data evaluation on ligature sets.

5.5.4 Comparison to Previous Research

The work on Urdu OCR is quite recent. The lack in the research for Urdu OCR has been mainly due to the unavailability of standard Urdu datasets, context sensitive and cursive nature of the Urdu script [3]. The research started on Urdu OCR in mainly the 2000s and that too with only the recognition of isolated Urdu characters. The recognition of isolated Urdu characters has been done with a higher recognition accuracy in the past with achieving near 100% accuracy. But the word and ligature level classification are still way behind. The present research on Urdu OCR has shifted towards the holistic approach for classification where the unit of recognition is mostly ligatures due to the text segmentation issues [1][3][11].

Proposed Method:
Our proposed method has also applied deep learning over our synthetically generated CEFAR dataset for Urdu ligature recognition. In our research, we first created the CEFAR dataset that contained exhaustive combinations of 2 and 3 character length ligatures for Urdu language. The recognition of these large number of ligature images, having the challenge of low intra-class variability and high number of

classes, was solved using the deep learning method. We used RNNs with architecture of ResNet34 containing 34 convolutional layers followed by a fully connected layer for classification. Our novel method reduced the class intra-class variability by data augmentation and redefinition of classes before training the model to enhance the network training and to reduce the effect of overfitting and thereby achieved an overall accuracy of 97.5% which is far more than the existing methods for Urdu ligature classification using deep learning.

Previous Research work:
Quite a lot of research has been carried out for Urdu script recognition. The focus here are the research works carried out for holistic Urdu ligature classification using deep learning. Few notable contributions are discussed in the section below;

In [12], Bidirectional Short Long Term (BLSTM) networks which were a variant of the Recurrent Neural Networks (RNNs) were used for the recognition of Urdu ligatures. The datasets used were Urdu Jang Dataset and UCOM dataset. This research worked on 2 different variations of the dataset. One ignoring the position of character within the ligature and other by considering the position information. The model can predict with an accuracy of 88.94% while ignoring position and with 88.79% accuracy while considering position information.

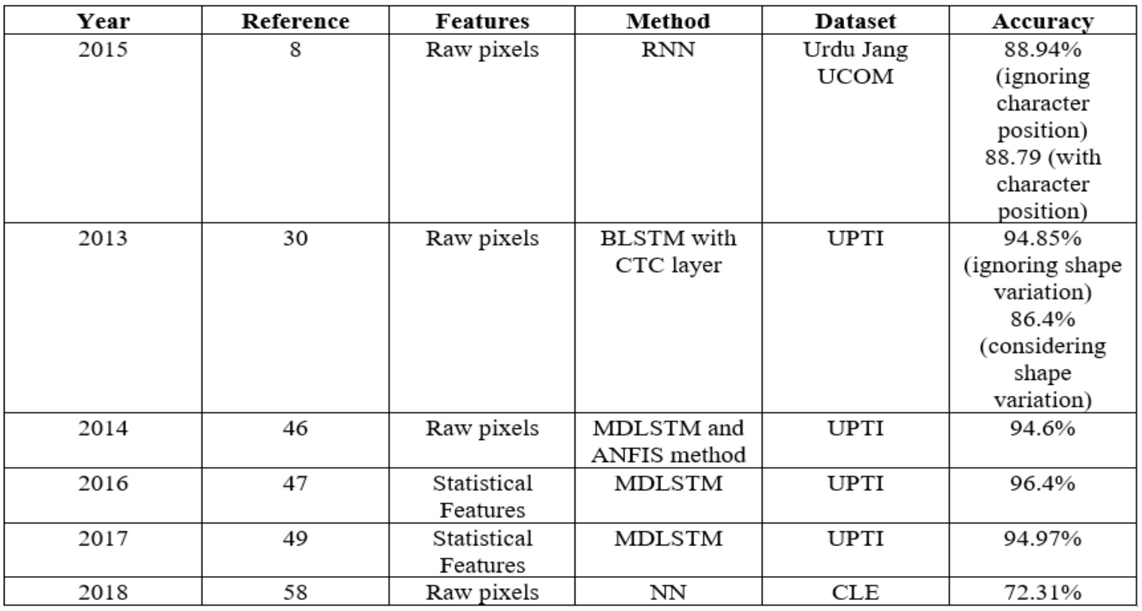
In another study, [30], printed Urdu text was recognized using BLSTM networks using CTC layer. The dataset used for evaluation was UPTI dataset. The model predicted with an accuracy of 94.85% at the character level while ignoring the shape variation and 86.4% accuracy while considering shape variation.

[46] used Multi-Dimensional LSTM (MDLSTM) network and ANFIS method for recognition of Urdu script. The model recognized the UPTI dataset with an accuracy of 94.6%.

In [47], the statistical features were extracted using an overlapping sliding window. These features were then fed to the MDLSTM network for classification. The technique classified the UPTI dataset with an accuracy of 96.4%.

[49] used the method of extracting statistical features and utilizing the memory of MDLSTM networks to classify the ligatures using UPTI dataset with an accuracy of 94.97%.

[58] used a segmentation free approach to recognize Urdu ligatures. 17,010 ligatures were utilized from the CLE Database. The proposed technique first applies clustering for reducing the number of classes and then used a deep neural network with dropout regularization for classification. The method achieved an accuracy of 72.31%.



Year	Reference	Features	Method	Dataset	Accuracy
2015	8	Raw pixels	RNN	Urdu Jang UCOM	88.94% (ignoring character position) 88.79 (with character position)
2013	30	Raw pixels	BLSTM with CTC layer	UPTI	94.85% (ignoring shape variation) 86.4% (considering shape variation)
2014	46	Raw pixels	MDLSTM and ANFIS method	UPTI	94.6%
2016	47	Statistical Features	MDLSTM	UPTI	96.4%
2017	49	Statistical Features	MDLSTM	UPTI	94.97%
2018	58	Raw pixels	NN	CLE	72.31%

Table 5.6: Summary of previous research work for Urdu ligature recognition

Our proposed method outperforms all the previous methods for Urdu ligature classification with a remarkable accuracy of 97.5% for both 2 and 3 character ligature set.

Chapter 6

Conclusion and Future Directions

This chapter summarizes the outcomes of the thesis. It explains the future work and research gap along with the research objectives and how these objectives were achieved from the proposed research method. Not only this chapter also entails the limitations of this work and the future directions.

6.1 Future Work

This thesis focuses on the classification of Urdu ligatures using an exhaustive combinations of Urdu characters. The dataset covers the ligatures of length 2 and 3. In future, this study can be extended to include ligatures of length longer than 3 and classification can be performed on them to create more robust OCR systems.

6.2 Research Gap

Research in the field of OCR for Urdu language has not gained much attention in the past. The reason for it has been the absence of a benchmark dataset for Urdu. Urdu is a cursive language; its writing style is highly cursive and varies with the writing style. Moreover, the context sensitive nature of this language makes it another big challenge. A single character can have multiple shapes based on its position in the word. Due to all these previously stated issues, no benchmark dataset could be created for Urdu language. Many synthetic and handwritten datasets have been made in an attempt to create a benchmark dataset but none could be considered as a standard Urdu dataset [3][6][11].

This research was carried out to fill this gap by providing a benchmark dataset for Urdu and also to apply deep learning to classify the generated dataset for Urdu

OCR. Classification and recognition are the major tasks in an OCR system. Many OCRs for English and Arabic language have been made but research in the field of Urdu OCR is new [3][6]. The existing methods for the recognition of Urdu language have mainly focused on the recognition of isolated Urdu characters or segmentation based methods for ligature or text recognition. The inaccuracy of these methods has shifted the research horizon towards holistic approach of Urdu ligature recognition using deep learning. The existing deep learning methods for Urdu recognition do not classify with high accuracy. This research work focuses on solving this challenge of Urdu ligature recognition and absence of benchmark dataset.

6.3 Research Objectives

Following research objectives were achieved by conducting this research.

6.3.1 Benchmark Dataset

It has previously been mentioned that there is an absence of a benchmark dataset for Urdu language because of which no notable and reliable research could be done in the field of Urdu OCR [6][11]. A benchmark dataset is the first requirement for any machine learning or deep learning problem. CEFAR dataset had been generated in the supervision of Dr Khawar Khurshid and his team. This dataset contained exhaustive combination of Urdu characters of length 2 and 3. This dataset contained an extensive number of images having 454720 ligatures. This dataset is an attempt to create a benchmark dataset for Urdu.

6.3.2 Urdu Script recognition

The major objective of this research was to create a classification system for Urdu language. The classification methods in the previous studies have attempted to classify isolated Urdu characters or the classification of word, ligature or line by applying segmentation, Segmentation based methods have many issues and are not reliable. These methods did not perform well in terms of classification accuracy. Recently, the research focus has been moved to holistic recognition of Urdu script using deep learning, where the recognition unit is ligature instead of word or text line. Research in this area is new and there is a lot of room for improvement. Some research studies have used different deep learning techniques for classification of Urdu ligatures but the performance of these deep learning models is not very high. Our research has focused on classification of ligatures in our generated CEFAR dataset.

6.3.3 Data with Low Intra-Class Variability and High Number of Classes to be recognized

Our synthetically generated CEFAR dataset contained large number of ligature images but one major challenge with having such a large number of image data was the low intra-class variability of the data. This dataset had unique Urdu ligatures and due to presence of a lot of unique ligatures, there were equally high number of classes. The ligatures were unique, they formed separate classes but the number of representative samples within each class became very small. This led to the scarcity of features for representation of a class. This research has also devised a method to cater this problem of low intra- class variability and high number of classes.

6.4 Notable Contributions

Our proposed method is a step forward in the research of Urdu OCR and deep learning. Our proposed method solves the research problems with unique solutions. Following are the notable contributions of our research work and to fulfill the research objectives.

• Generation of the benchmark dataset is the foremost requirement in the field of Urdu OCR. This research generated the CEFAR dataset that is a step forward towards creating a standard dataset for Urdu.

• Data augmentation and class redefinition techniques solved the low-intra class variability and issue of high number of classes of the generated CEFAR dataset.

• This research not only focused on creating and augmenting the dataset but also the classification of Urdu ligatures using deep learning. We have used the Recurrent Neural Networks for the classification of Urdu ligatures which achieved the recognition accuracy of 97.15%, better than the existing deep learning methods for Urdu ligature recognition.

6.5 Limitations and Future Directions

Research studies can always be extended and improved. Likewise, our research also has the room for improvement. This research though covers a lot of areas including dataset generation, data augmentation, reduction and classification. Yet, some limitations still exist. The dataset generated contains ligatures of 2 and 3 character length. Ligatures could be generated for length of 4 or more. Generation and classification could be the same as proposed in our research. In future, more robust methods could be created to classify the Urdu ligatures with even more accuracy than the one achieved by using the proposed method.

References

[1] Manoj krishna, M., Neelima, M., Harshali, M. and Venu Gopala Rao, M., 2018. Image classification using Deep learning. International Journal of Engineering Technology, 7(2.7), p.614.

[2] Nath, S., Mishra, G., Kar, J., Chakraborty, S. and Dey, N., 2014. A survey of image classification methods and techniques. 2014 International Conference on Control, Instrumentation, Communication and Computational Technologies (ICCICCT).

[3] Khan, N. and Adnan, A., 2018. Urdu Optical Character Recognition Systems: Present Contributions and Future Directions. IEEE Access, 6, pp.46019-46046.

[4] Mohammad, F., Anarase, J., Shingote, M. and Ghanwat, P., 2014. Optical Character Recognition Implementation Using Pattern Matching. (IJCSIT) International Journal of Computer Science and Information Technologies, Vol. 5 (2), pp.2088-2090.

[5] Baker, P., Hardie, A., McEnery, T., Cunningham, H., Gaizauskas, R. (2002). EMILLE, a 67-million word corpus of Indic languages: data collection, mark-up and harmonization. In Proceedings of LREC 2002 (pp. 819-825) https://www.aclweb.org/anthology/L02-1319/

[6] Lehal, G., 2012. Choice of recognizable units for URDU OCR. Proceeding of the workshop on Document Analysis and Recognition - DAR '12,

[7] Kaur, G., Singh, S. and Kumar, A., 2017. Urdu Ligature Recognition Techniques- A Review. International Conference on Intelligent Communication and Computational Techniques (ICCT)

[8] Ahmed, S., Naz, S., Razzak, M., Rashid, S., Afzal, M. and Breuel, T., 2015. Evaluation of cursive and non-cursive scripts using recurrent neural networks.

Neural Computing and Applications, 27(3), pp.603-613.

[9] Husain, S., 2002. A multi-tier holistic approach for Urdu Nastaliq recognition. International Multi Topic Conference, 2002. Abstracts. INMIC 2002.

[10] Mikolajczyk, A. and Grochowski, M., 2018. Data augmentation for improving deep learning in image classification problem. 2018 International Interdisciplinary PhD Workshop (IIPhDW).

[11] Ud Din, I., Siddiqi, I., Khalid, S. and Azam, T., 2017. Segmentation-free optical character recognition for printed Urdu text. EURASIP Journal on Image and Video Processing, 2017(1).

[12] Naz, S., Hayat, K., Imran Razzak, M., Waqas Anwar, M., Madani, S. and Khan, S., 2014. The optical character recognition of Urdu-like cursive scripts. Pattern Recognition, 47(3), pp.1229-1248.

[13] Ijaz, M., Hussain, S, 2007. Corpus Based Urdu Lexicon Development.

[14] Urooj, Saba Hussain, Sarmad Adeeba, Farah Jabeen, Farhat Perveen, Raheela. ,2012. CLE Urdu Digest Corpus.

[15] Slimane, F., Ingold, R., Kanoun, S., Alimi, A. and Hennebert, J., 2009. A New Arabic Printed Text Image Database and Evaluation Protocols. 2009 10th International Conference on Document Analysis and Recognition,.

[16] Naz, S., Umar, A., Ahmed, R., Razzak, M., Rashid, S. and Shafait, F., 2016. Urdu Nasta'liq text recognition using implicit segmentation based on multi-dimensional long short term memory neural networks. SpringerPlus, 5(1).

[17] Islam, Noman Islam, Zeeshan Noor, Nazia. ,2016. A Survey on Optical Character Recognition System. ITB Journal of Information and Communication Technology.

[18] Khorsheed, M., 2002. Off-Line Arabic Character Recognition – A Review. Pattern Analysis Applications, 5(1), pp.31-45.

[19] Pansare, S., Joshi, D. ,2014. A Survey on Optical Character Recognition Techniques.

[20] Mehran, R., Pirsiavash, H. and Razzazi, F., 2005. A Front-End OCR for

Omni-Font Persian/Arabic Cursive Printed Documents. Digital Image Computing: Techniques and Applications (DICTA'05),.

[21] Shamsher, I Ahmad, Zaheer Orakzai, J Adnan, Awais., 2007. OCR for printed Urdu script using feed forward neural network. the Proceedings of World Academy of Science, Engineering and Technology. 23.

[22] Khan, K., Ullah, R., Ahmad Khan, N. and Naveed, K., 2012. Urdu Character Recognition using Principal Component Analysis. International Journal of Computer Applications, 60(11), pp.1-4.

[23] Akram, Q., Hussain, S. and Habib, Z., 2013. Font Size Independent OCR for Noori Nastaleeq.

[24] Tariq, J., Nauman, U. and Umair Naru, M., 2010. Softconverter: A novel approach to construct OCR for printed Urdu isolated characters. 2010 2nd International Conference on Computer Engineering and Technology,.

[25] Tabassam, Nawaz Naqvi, Syed Rehman, Habib Anoshia, Faiz., 2009 . Optical Character Recognition System for Urdu (Naskh Font) Using Pattern Matching Technique. International Journal of Image Processing. 3.

[26] Ahmad, Z., Orakzai, J. and Shamsher, I., 2009. Urdu compound Character Recognition using feed forward neural networks. 2009 2nd IEEE International Conference on Computer Science and Information Technology,.

[27] Devi, D. and Ashifa, J., 2017. An Analysis of Urdu Optical Character Recogition Processing. International Journal of Information Technology (IJIT), [online] Volume 3(Issue 2).

[28] Husain, S., 2002. A multi-tier holistic approach for Urdu Nastaliq recognition. International Multi Topic Conference, 2002. Abstracts. INMIC 2002.,.

[29] Hussain, S., Ali, S. and Akram, Q., 2015. Nastalique segmentation-based approach for Urdu OCR. International Journal on Document Analysis and Recognition (IJDAR), 18(4), pp.357-374.

[30] Ul-Hasan, A., Ahmed, S., Rashid, F., Shafait, F. and Breuel, T., 2013. Offline Printed Urdu Nastaleeq Script Recognition with Bidirectional LSTM Networks. 2013 12th International Conference on Document Analysis and Recognition,.

[31] Tariq, J., Nauman, U. and Umair Naru, M., 2010. Softconverter: A novel approach to construct OCR for printed Urdu isolated characters. 2010 2nd International Conference on Computer Engineering and Technology,.

[32] Khan, E. and Khan, E., 2015. Urdu Optical Character Recognition Technique for Jameel Noori Nastaleeq Script. Journal of Independent Studies and Research - Computing, 13(1).

[33] Pal, U. and Sarkar, A., n.d. Recognition of printed Urdu script. Seventh International Conference on Document Analysis and Recognition, 2003. Proceedings.,.

[34] Khan, W. and Khan, R., 2015. Urdu optical character recognition technique using point feature matching; a generic approach. 2015 International Conference on Information and Communication Technologies (ICICT),.

[35] Ud Din, I., Siddiqi, I., Khalid, S. and Azam, T., 2017. Segmentation-free optical character recognition for printed Urdu text. EURASIP Journal on Image and Video Processing, 2017(1).

[36] Singh, P., Sarkar, R., Das, N., Basu, S. and MitaNasipuri, 2013. Identification of Devnagari and Roman Scripts from Multi-script Handwritten Documents. Lecture Notes in Computer Science, pp.509-514.

[37] Naz, S., Hayat, K., Imran Razzak, M., Waqas Anwar, M., Madani, S. and Khan, S., 2014. The optical character recognition of Urdu-like cursive scripts. Pattern Recognition, 47(3), pp.1229-1248.

[38] Mushtaq, F., Misgar, M., Kumar, M. and Khurana, S., 2021. UrduDeepNet: offline handwritten Urdu character recognition using deep neural network. Neural Computing and Applications, 33(22), pp.15229-15252.

[39] Hussain, S., Zaman, S. and Ayub, M., 2009. A Self Organizing Map based Urdu Nasakh character recognition. 2009 International Conference on Emerging Technologies,.

[40] Habib Khan, N., Adnan, A., Waheed, A., Zareei, M., Aldosary, A. and Mahmoud Mohamed, E., 2021. Urdu Ligature Recognition System: An Evolutionary Approach. Computers, Materials Continua, 66(2), pp.1347-1367.

[41] Mir, S., Zaman, S. and Anwar, M., 2015. Printed Urdu Nastalique Script Recognition Using Analytical Approach. 2015 13th International Conference on

Frontiers of Information Technology (FIT),.

[42] Arica, N. and Yarman-Vural, F., 2001. An overview of character recognition focused on off-line handwriting. IEEE Transactions on Systems, Man and Cybernetics, Part C (Applications and Reviews), 31(2), pp.216-233.

[43] Naz, S., Umar, A., Ahmad, R., Ahmed, S., Shirazi, S. and Razzak, M., 2015. Urdu Nasta'liq text recognition system based on multi-dimensional recurrent neural network and statistical features. Neural Computing and Applications, 28(2), pp.219-231.

[44] Khan, K., Khan, R., Alkhalifah, A. and Ahmad, N., 2015. Urdu text classification using decision trees. 2015 12th International Conference on High-capacity Optical Networks and Enabling/Emerging Technologies (HONET),.

[45] Khan, K., Ullah, R., Ahmad Khan, N. and Naveed, K., 2012. Urdu Character Recognition using Principal Component Analysis. International Journal of Computer Applications, 60(11), pp.1-4.

[46] Rinku Patel , Mitesh Thakkar. R."Handwritten Nastaleeq Script Recognition with BLSTM-CTC and ANFIS method". International Journal of Computer Trends and Technology (IJCTT) V11(3):131-136, May 2014. ISSN:2231-2803

[47] Naz, S., Umar, A., Ahmad, R., Ahmed, S., Shirazi, S., Siddiqi, I. and Razzak, M., 2016. Offline cursive Urdu-Nastaliq script recognition using multidimensional recurrent neural networks. Neurocomputing, 177, pp.228-241.

[48] Naz, S., Umar, A., Ahmad, R., Siddiqi, I., Ahmed, S., Razzak, M. and Shafait, F., 2017. Urdu Nastaliq recognition using convolutional–recursive deep learning. Neurocomputing, 243, pp.80-87.

[49] Abramovich, F. and Pensky, M., 2019. Classification with many classes: Challenges and pluses. Journal of Multivariate Analysis, 174, p.104536.

[50] Johnson, J. and Khoshgoftaar, T., 2019. Survey on deep learning with class imbalance. Journal of Big Data, 6(1).

[51] Buda, M., Maki, A. and Mazurowski, M., 2018. A systematic study of the class imbalance problem in convolutional neural networks. Neural Networks, 106, pp.249-259.

[52] Weiss, K., Khoshgoftaar, T. and Wang, D., 2016. A survey of transfer learning. Journal of Big Data, 3(1).

[53] Hasan Finjan, R., Salim Rasheed, A., Abdulsahib Hashim, A. and Murtdha, M., 2021. Arabic handwritten digits recognition based on convolutional neural networks with resnet-34 model. Indonesian Journal of Electrical Engineering and Computer Science, 21(1), p.174.

[54] A Paszke, S Gross, F. Massa, A Lerer, J Bradbury, G Chanan, T Killeen, Z Lin, N Gimelshein, L Antiga, A Desmaison, A Köpf, E Yang, Z DeVito, M Raison, A Tejani, S Chilamkurthy, BSteiner, L Fang, J Bai, and S Chintala., 2019. PyTorch: an imperative style, high-performance deep learning library. Proceedings of the 33rd International Conference on Neural Information Processing Systems. Curran Associates Inc., Red Hook, NY, USA, Article 721, 8026–8037.

[55] Huang, C., Li, Y. and Yao, X., 2020. A Survey of Automatic Parameter Tuning Methods for Metaheuristics. IEEE Transactions on Evolutionary Computation, 24(2), pp.201-216.

[56] M, H. and M.N, S., 2015. A Review on Evaluation Metrics for Data Classification Evaluations. International Journal of Data Mining Knowledge Management Process, 5(2), pp.01-11.

[57] Ud Din, I., Siddiqi, I., Khalid, S. and Azam, T., 2017. Segmentation-free optical character recognition for printed Urdu text. EURASIP Journal on Image and Video Processing, 2017(1).

[58] Naz, S., Umar, A., Ahmad, R., Ahmed, S., Shirazi, S. and Razzak, M., 2015. Urdu Nasta'liq text recognition system based on multi-dimensional recurrent neural network and statistical features. Neural Computing and Applications, 28(2), pp.219-231.

[59] Rafeeq, M., ur Rehman, Z., Khan, A., Khan, I. and Jadoon, W., 2018. Ligature categorization based Nastaliq Urdu recognition using deep neural networks. Computational and Mathematical Organization Theory, 25(2), pp.184-195.

Appendix: Software and System Specifications

Software Specifications:

• The core programming language used in this research was python (https://www.python.org/).

• The jupyter notebook server (https://jupyter.org/)was used for writing code and training the network.

• Fastai library (https://www.fast.ai/)was used for performing deep learning tasks.

• pytorch (https://pytorch.org/) is an open-source machine learning framework used to perform various machine and deep learning tasks, was used for running fastai for downloading and training the Renet34 architecture.

• Evaluation of the trained deep learning model and get the evaluation metrices and visualization using graphs and plots , the python libraries of sci-kit (https://pypi.org/project/scikit-learn/) and matplotlib (https://matplotlib.org/) were used respectively.

• Numpy library (https://numpy.org/)was used to deal with arrays, matrices and to perform complex mathematical problems.

System Specifications:

The deep learning model training and evaluation was performed on the GPU system with following system specifications;

• NVidia Tesla T4 GPU server

The GPU facility was granted by the NUST Interdisciplinary Cluster for Higher Education (NICHE) at NUST Islamabad.