Exploration of Deep Learning Based Recognition for Urdu Text

Sumaiya Fazal Dad

Master of Sciences

A thesis submitted in partial fulfillment
of the requirements for the degree of *Master of Science*
at the *National University of Computer and Emerging Sciences*

Department of Computer Science

**National University of Computer and Emerging Sciences,
Peshawar, Pakistan**

2018

Plagiarism Undertaking

Sumaiya Fazal Dad
Dated:

Verified by Plagiarism Cell Officer
Dated:

Author's Declaration

I, Sumaiya Fazal Dad, hereby state that my MS thesis titled "Exploration of Deep Learning Based Recognition for Urdu Text" is my own work and it has not been previously submitted by me for taking partial or full credit for the award of any degree at this University or anywhere else in the world. If my statement is found to be incorrect, at any time even after my graduation, the University has the right to revoke my MS degree.

Sumaiya Fazal Dad
Dated:

Certificate of Approval

It is certified that the research work presented in this thesis, titled: "*Exploration of Deep Learning Based Recognition for Urdu Text*", was conducted by *Sumaiya Fazal Dad* under the supervision of *Dr. Mohammad Nauman*. No part of this thesis has been submitted anywhere else for any other degree. The thesis is submitted to the Department of Computer Science in partial fulfillment of the requirements for the degree of *Master of Science in Computer Science* at the National University of Computer and Emerging Sciences, Peshawar, Pakistan in August, 2018.

Candidate

Sumaiya Fazal Dad	Signature: ________________

Internal Examiner

Dr. Nauman Azam	Signature: ________________

External Examiner

Dr. Naveed Islam Islamia College University, Peshawar	Signature: ________________

____

Dr. Usman Habib

Graduate Program Coordinator
National University of Computer and Emerging Sciences, Peshawar

____

Dr. Omar Usman Khan

HoD of Computer Science
National University of Computer and Emerging Sciences, Peshawar

I dedicate this thesis to my mom who supported me in the entire process.

Acknowledgements

All the praise to ALLAH that induced the man with intelligence, knowledge and wisdom. Peace and blessing of Allah be upon the Holy Prophet who exhort his followers to seek for knowledge from cradle to grave. It is great pleasure to acknowledge individual and organizational help, which made this research possible.

Sumaiya Fazal Dad

16p-7011

Abstract

Urdu is national and official language of Pakistan. Which ultimately opens up the future direction for the research community to digitize Urdu language and develop an effective system to deal with OCR challenges faced by Urdu. Resource scarcity exists in Urdu language, in spite of the fact, that it is widely spoken in South Asian countries as well as some parts of Europe. To digitize a language, character segmentation is an essential yet critical area for many languages. In Eastern language, character classification can be ruthlessly hard as compared to its western counterparts.

Urdu is a cursive script language and have similarities with Arabic and many other South Asian languages. Urdu is more complex to classify due to its complex geometrical and morphological structure. Character classification can be processed further if segmentation technique is efficient, but due to context sensitivity in Urdu, segmentation-based recognition often results with high error rate. However, character classification has implemented for the many western languages, but Urdu research community have shifted their focus to the ligature recognition as it takes minimal segmentation.

Our proposed approach for Urdu optical character recognition system is a component-based classification relying on automatic feature learning. It is a technique called convolutional neural network, a deep learning architecture, replaces a manually engineered features by implementing shared weights of a network. CNN is trained and tested on Urdu text dataset, which is generated through permutation process of three characters and further proceeds to discarding unnecessary images by applying connected component technique in order to obtain ligature only. Hierarchical neural network is implemented to deal with three degrees of character permutations. Level-0 outputs number of characters a component comprises and are being classified at next levels, for which it moves for the individual component recognition to the next levels depending on the component class of previous level. Our model successfully achieved 0.99% for component classification.

Contents

1 Preliminaries and Introduction **1**

1.1 Optical Character Recognition 1

1.2 Urdu Language Limitations 2

1.3 Automatic Learning 3

1.4 Roadmap 3

1.5 Thesis Organization 4

2 Review of Literature **5**

2.1 History of OCR 5

2.1.1 First Generation OCR 6

2.1.2 Second Generation OCR 6

2.1.3 Third Generation OCR 6

2.2 Pipeline of OCR 7

2.2.1 Optical Scanning 7

2.2.2 Location and Segmentation 7

2.2.3 Pre-Processing 7

2.2.4 Feature Extraction 8

2.2.5 Classification 8

2.3 Complexities of Urdu Language 8

2.4 Segmentation Based Approaches for other Scripts 10

2.4.1 English Language Segmentation Algorithm 10

2.4.2 Chinese Language Segmentation Algorithm 11

2.5 Segmentation Based Approaches for Urdu Scripts 12

2.5.1 Pre-Processing 13

2.6 Feature Learning for Arabic Script 13

2.6.1 Feature Learning through CNN 14
2.7 Feature Learning for Pushto Script 15
2.8 Feature Learning for Urdu Scripts 15
2.8.1 Automatic Feature Learning 18
2.9 Connected Component labeling Algorithms 21
2.9.1 Stroke Width Transformation 21
2.9.2 Two Pass Connected Component Algorithm 22
2.10 Machine Learning 22
2.10.1 Linear Regression 22
2.10.2 Gradient Descent 24
2.10.3 Stochastic Gradient Descent 25
2.10.4 Classification 25
2.10.4.1 Cost Function 26
2.10.4.2 Regularization 26
2.11 Artificial Neural Network and Deep Learning 27
2.11.1 Biological Motivation 27
2.11.2 Perceptron 27
2.11.3 Back Propagation Algorithm: 28
2.11.4 Supervised Learning 30
2.11.4.1 Activation Function 30
2.11.4.2 Error Measure 30
2.11.4.3 Training Protocols 30
2.11.4.4 Optimization Technique 31
2.11.5 Regularization 31
2.12 Deep Learning 32
2.13 Convolutional Neural Network 33
2.13.1 ConvNets Architecture 34
2.13.2 Residual Nets 37

3 Proposed Method **38**
3.1 Dataset Generation 38

3.1.1 Component Based Identification 38

3.2 Convolutional Neural Network Architecture 40

3.3 Hierarchical neural network 42

4 Results **44**

4.1 Level-0 44

4.2 Level-1 45

4.2.1 Degree-1 45

4.2.2 Degree-2 46

4.2.3 Degree-3 47

4.3 Summary of Results 47

4.3.1 Dataset Specification 47

4.3.2 CNN Architecture Configuration 48

4.3.3 Recognition Rate 49

4.3.4 Model Summary 49

5 Conclusions and Future Work **52**

List of Figures

1.1 Complexities of Nastalique writing styles [1] 2

2.1 Components of OCR 8
2.2 Shape Variation [2] 9
2.3 a. Inter ligature overlapping b. Intra ligature overlapping [2] 9
2.4 Segmentation Algorithm [3] 11
2.5 Space Omission Problem 12
2.6 Space Insertion Problem 13
2.7 Arabic word, Connected Component and Group [4] 14
2.8 A typical CNN architecture for proposed CNN for offline Arabic handwriting recognition [5] 15
2.9 Nuqtas, Curve points and End points [6] 18
2.10 Convolution-Recursive Deep learning model [7] 19
2.11 GBLSTM Network[8] 20
2.12 Nabocr [9] 21
2.13 Gradient Descent [10] 24
2.14 Perceptron [11] 28
2.15 Back Propagation Algorithm [12] 29
2.16 (a) Neural Network before adding dropout (b) Neural Network after dropout [13] 32
2.17 Feature Extraction [14] 33
2.18 ConvNet Layers 34
2.19 Max Pooling 36
2.20 Resiudal Nets [15] 37

3.1 Block Diagram for Recognition System 39

3.2 Connected Component 40
3.3 Dataset Creation 41
3.4 Hierarchical Neural Networks 43

4.1 Level-0 45
4.2 Degree-1 at Level-1 46
4.3 Degree-2 at Level-1 46
4.4 Degree-3 at Level-1 47

5.1 Model of Proposed Future Work 54

List of Tables

2.1 Types of ligatures and corresponding recognition rates [16] 17
2.2 Notable studies on Urdu OCR with recognition rates [16] 17

4.1 Dataset Specification 48
4.2 CNN Architecture Configuration Part-1 48
4.3 CNN Architecture Configuration Part-2 48
4.4 Recognition Rate 49
4.5 CNN Model Summary Level-0 50
4.6 CNN Model Summary Level-1 for Degree-1 50
4.7 CNN Model Summary Level-1 for Degree-1 51
4.8 Residual CNN Model Summary at Level-1 for Degree-3 51

Chapter 1

Preliminaries and Introduction

Urdu is national language of Pakistan and recently declared as official language. It is a popular and widely spoken language of Pakistan, India, Bangladesh and other South Asian countries along UK and Canada. Almost, 11 million speakers of Urdu are in Pakistan and 300 million and above are in the entire world [17]. Urdu follows two different writing styles Naskh and Nastaliq. Urdu writing is based on Nastaliq style however, Arabic is based on Naskh style of writing which is different from Nastaliq in terms of diagonality [18]. Urdu is rather complex as its morphology and syntax structure foundations are mixture of Persian, English, Turkish, Sanskrit and Arabic [19].

1.1 Optical Character Recognition

Optical character recognition (OCR) process is based on scanning document and converting it into a text file to make it editable but OCR process will produce more efficient results if words or characters are defined by some boundary. Character segmentation is a critical area in OCR due to the fact that isolated characters have high recognition as compared to character in a word. OCR segmentation is further divided into three categories, classical approach, recognition-based segmentation and holistic approach. Space at the end of a word defines a boundary but not all languages follow rule or agreement for defining a word boundary and Urdu is one of those languages.

1.2 Urdu Language Limitations

Field of natural language processing is transformed in recent years and newly developed systems produced some phenomenal results in the field of image recognition. Western languages are ahead of deploying updated systems as they are rich-resource languages due to the wide range of work performed in past, while same is not true when it comes to South Asian languages, resource scarcity is one of the major setback behind a slow research in Urdu OCR as not enough data is available for training purpose, and Urdu language also have more limitations and complexities of defining a word or character as compared to any other language which imposes a barrier to find a suitable technique. Challenges and complexities (See Figure.1.1) in implementation of Urdu Nastaliq recognition [20] are briefly explained in various papers.

Digitization of printed Urdu text takes input text and tokenizes it into words and characters, in case of English character, tokenization is an easy process because spaces at the end of word defines a boundary. There are many Asian languages which are probably similar such as Thai, Khmer, Lao and Dzongkha but did not offer word boundaries and therefore avoids the use of white spaces to mark word endings [21]. Segmentation in Urdu language is a complex procedure it faces space omission and space insertion error.

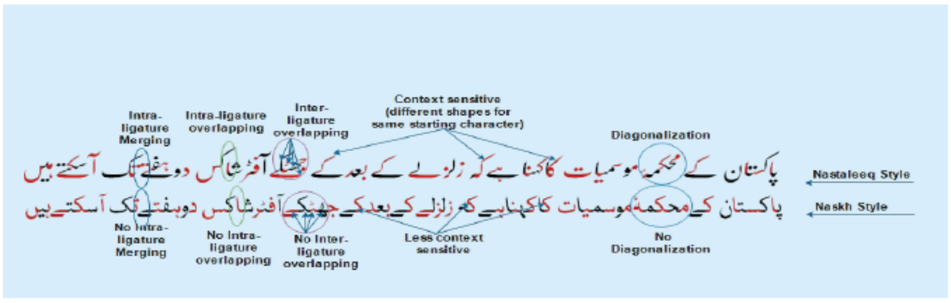

Intra-ligature Merging
Intra-ligature overlapping
Inter-ligature overlapping
Context sensitive (different shapes for same starting character)
Diagonalization
پاکستان کے محکمہ موسمیات کا کہنا ہے کہ زلزلے کے بعد کے جھٹکے آفٹر شاکس دو ہفتے تک آسکتے ہیں
Nastaleeq Style
پاکستان کے محکمہ موسمیات کا کہنا ہے کہ زلزلے کے بعد کے جھٹکے آفٹر شاکس دو ہفتے تک آسکتے ہیں
Naskh Style
No Intra-ligature Merging
No Intra-ligature overlapping
No Inter-ligature overlapping
Less context sensitive
No Diagonalization

Figure 1.1: Complexities of Nastalique writing styles [1]

1.3 Automatic Learning

Character and ligature recognition of printed Urdu documents passes through segmentation procedure first and segmentation of Urdu is not only a complex procedure, the segmentation based recognition often results with high error rate, recognizing a whole word in Urdu OCR is yet unaccomplished, due to this reason mostly researches are moving to the segmentation free methods where recognition is apparently easy, character, ligature and sentence based recognitions are providing good results and with diversity in innovative research the researchers have shift focus on ligature and sentence based OCR systems. Ligature recognition through feature learning produced some promising results but manually hand-engineered features requires domain expertise and must need to be carefully designed. A prevalent feature extractor technique for better learning needs to be used to obtain high results.

1.4 Roadmap

Our proposed approach of optical character recognition system is based on automatic feature learning of Urdu components. Urdu is facing number of limitations while implementing segmentation based recognition due to no language agreement of Urdu characters. For instance a character shape varies depending on its position in a word. Space insertion and space omission problem in a word makes segmentation difficult and without segmenting a character or word recognition is complex. Due to these limitations the research trend is moving toward ligature recognition where a segmentation of word or character is not necessary and results in low error rate. Our method performs recognition based on segmentation free ligature but we also narrowed down the ligature to component only where all the dots and diacritics are removed and only common shape characters are left. The dataset for training is generated through a permutation of characters. Our dataset for proposed solution is limited to three characters only because most of the Urdu characters are combination of two or three characters, in rare cases we will find word reached to seven or eight characters. Generated dataset is then trained through convolutional neural network and extended on two levels through hierarchical neural network. Result we achieved reports accuracy based on convolutional neural network training.

1.5 Thesis Organization

Chapter one is an introductory chapter contains the short explanations of the main topics of our research. Initially it gives the overview of optical character recognition of how the process works, OCR segmentation and types. Another section provides overview and history of our problem statement through examples and a need to overcome such limitations. An important section leading towards the solution of segmentation free recognition through automatic feature learning technique which is preferred rather than hand-crafted features due to certain limitation and how it is producing better results than state-of-the-art methods. At the end roadmap has briefly discussed the problem statement, scope of our research and thesis contribution.

Chapter two is literature review section starts with history of OCR through out ages with explanation of procedures of OCR. Complexities of Urdu OCR are briefly discussed along examples. Due to high error rate in segmentation Urdu research is moved towards feature learning for that reason some review of how feature learning have produced good results for other scripts is explained. Pattern recognition starts with a machine learning concepts, to give an idea about machine learning basics it is discussed in different section and further extended to the convolutional neural network which is our main technique.

Chapter three is proposed approach and is briefly explained in different sections, starts with the dataset creation and how component based identification is taking place, details of convolutional neural network architecture and levels of hierarchical neural network for two different levels are briefly discussed in this chapter.

Chapter four result chapter holds discussion of each dataset, we have total four datasets containing almost 22,000 to 66,000 images each dataset is trained on convolutional neural network. CNN model summary for each dataset, graphical representation of loss and accuracy is explained in detail.

Chapter 2

Review of Literature

OCR is the process where scanned document is converted into a text file this procedure works more effectively where characters and word boundaries are defined. Character segmentation has long been a critical area of the OCR process. The higher recognition rates for individual or isolated characters to those achived for words and connected components are far better [22]. Segmentation strategies are divided into three groups first is classical approach which is mainly based on dissecting or cutting image into understandable components if it fulfills all the required properties of a character, second segmentation strategy is recognition based and third is holistic method.

2.1 History of OCR

Mimicking human behavior through AI technologies is a researcher dream and improved with the time, first character recognition is originated from 1870 when the same year retina scanner invented which was able to perform image transmission, couple of decades later Polish P. Nipkow set major breakthrough in the field of optical character recognition by inventing a sequential scanner, but in 19th century OCR was experimented to benefit blind people to develop a guidance system. In 1940 invention of computer eventually broke the stagnant development circle and brought us an OCR application which are an important part of every corporation in todays world. In 1950 when large amount of data was unable to handle, OCR application made first appearance commercially, but 1954 is the year when typed sales reports where converted into punch cards so that it can be used as an input to the computer. In smart phone era now OCR can be used by taking a picture from camera and through available internet connection.

2.1.1 First Generation OCR

In 1960 first generation of OCR appeared, but the scope and functionality of machine was limited to the one font, a specially designed letter specifically for machine was used, later on, up to 10 fonts machines introduced into the market, pattern recognition methods were not robust enough which made character recognition limited, initially, character image library was implemented to compare both characters for a one font only. NOF, IBM 1418, Farrington 360 are examples of first generation OCR and they used special fonts only.

2.1.2 Second Generation OCR

In middle of 1960 and early 70s machines were improved enough to recognize printed characters as well as hand printed characters, the hand printed characters were limited to few letters along with numbers and symbols. First system was developed by IBM and later on Hitachi released first OCR system. Meanwhile OCR requirements were under consideration and completed in 1966 which produced American standard OCR character set, OCR-A, it was according to the requirements of OCR. IBM 1287, NEC, Toshiba are known as second generation OCR and used for handwritten alpha-numeric.

2.1.3 Third Generation OCR

Middle of 1970 focused was shifted to all kind of characters, hand-written, noisy and large printed but with high performance and low cost due to the advancement in hardware. Documents created on typewriter were easily modified on the computers as it offered small fonts and spacing was uniform. Examples of third generation OCR are IBM 1975, used upto 275 fonts and was able to recognized poorly hand-written words.

2.2 Pipeline of OCR

Optical character recognition consists of components, every level is input to the next level and forms a pipeline (See Figure. 2.1). First stage is digitization, converting a document into digital text through scanner device. Location of a text must be segmented and then extracted. The extracted text is pre-processed by removing noise and other limitations and to make it usable for machine, at the end stage classification is performed.

2.2.1 Optical Scanning

Scanning device is a device which digitizes the analogue document. It captures the digital image and thresholding is used to transform black and white document into grayscale image for easy processing. Fixed threshold level is defined in thresholding [23], threshold is a certain value, below the threshold gray levels are converted to black and above will remain white. Thresholding reduces memory and computational time.

2.2.2 Location and Segmentation

An efficient segmentation method can achieve high results, it is necessary stage of OCR process, the region of text must be located and extracted through segmentation. Segmentation can be performed for characters and words, whereas, segmented words are further dissected into characters. Printed text can be easily segmented because white spaces are easily locate-able but hand-written documents may cause hinder due to joined nature.

2.2.3 Pre-Processing

Noisy images such as gaps or distortion can lead to downgrading the performance of OCR and returns poor results, pre-processing of images is a compulsory step and eliminates all the defects up to a certain level. At first, binarize the image into black and white for

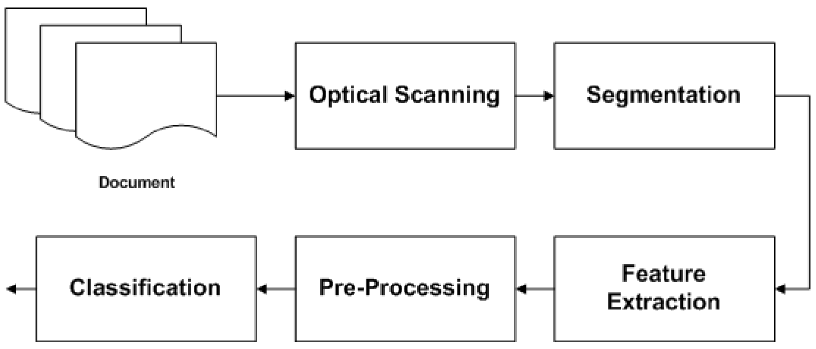


Figure 2.1: Components of OCR

fast processing, performing slant correction for hand-written words also normalizing can transform characters into uniform size and filling can fill the broken or thin lines, rotation can also be used to correct orientation.

2.2.4 Feature Extraction

To capture the underlying patterns of a character is a complex procedure of pattern recognition. Carefully hand-engineered features can extract such properties that will produce far better results than stat-of-the-art methods. Features are divided under different groups, features are extracted based on distribution of points and structure of character after feature extraction supervised learning algorithms assigns class to recognized features.

2.2.5 Classification

Feature recognition requires more training and domain expertise for manually construction of features, after recognition process classification is performed by using supervised learning algorithms, such as SVM and feed forward neural network or any hybrid approach.

2.3 Complexities of Urdu Language

Danish et al. [20] briefly explained complexities in Urdu are:

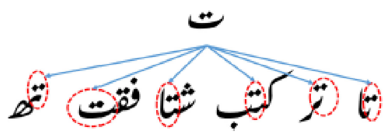


Figure 2.2: Shape Variation [2]

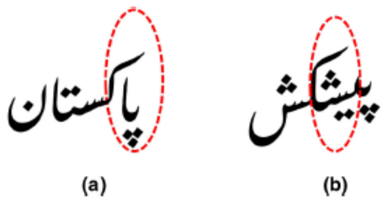


Figure 2.3: a. Inter ligature overlapping b. Intra ligature overlapping [2]

- Number of character shapes: In Arabic letter have 4 different shapes and in Urdu character shape highly depends on position and character to which it is being joined. (See Figure.2.2) represents shape variation.

- Slopping: Slopping means that when two letters merged together they create a slope, and this slope is called diagonality, these diagonal words are written from top-right to bottom-left. Now no such concept of diagonality exists in western language [24]. It also means no join of characters on base line. No segmentation algorithm can produce good result for sub-words in Urdu, even if algorithm succeeded for Arabic & Persian languages.

- Stretching: Letters change their standard positions by transforming into longer versions depending on the context [24]. Stretching is also close to context-sensitivity of Nastalique script, only certain type of position characters can join if they have certain type of class. It specially cause problems if we are dealing with calligraphic and handwritten Nastalique documents, machine printed documents does not create this problem.

- Positioning & Spacing: Placement of sub-words and ligatures is position, while space between ligature is spacing. In case of positioning ligature can be placed at various places despite of overlap between ligatures.

- Intra-ligature and Inter-ligature overlapping: In intra-ligature character within same ligature overlap each other, inter-ligature overlapping of individual characters from different sub-words vertically overlap each other. (See Figure. 2.3) represents intra-ligature and inter ligature.
- Filled Loop Characters: In Urdu, some characters have loops but due to Nastalique style inner filled loop can make recognition for machine extremely complex.
- False Loops: Character written in Nastalique joins the base which causes initial part of character to make false loop, to avoid this issue identify false loop and break them or recognize it without breaking.
- Varying Stroke Width: In Nastalique every character have different stroke width at different positions, it is an important part for recognition process, a word have primary and secondary component, stroke width removes secondary component and deals it at the end of recognition part while primary components are handled first.
- Complex Dot Placement Rules: In Urdu characters can have up to three dots but due to context sensitivity and positioning it can be moved nearby.

2.4 Segmentation Based Approaches for other Scripts

Optical character recognition is a pipeline consists of steps. First step is digitization of a document after optical scanning. Segmentation is a next step where character and word is segmented for recognition. Segmentation is suitable for languages where space is defined for example English there are few segmentation algorithms explained in this section which performed well in case of hand-written documents where characters are probably merged for many other languages such as English hand-written documents and Chinese.

2.4.1 English Language Segmentation Algorithm

M. Blumenstein et al. [25] proposed a segmentation algorithm for hand written words, initially involves pre-processing of an image, then segmentation points are located through

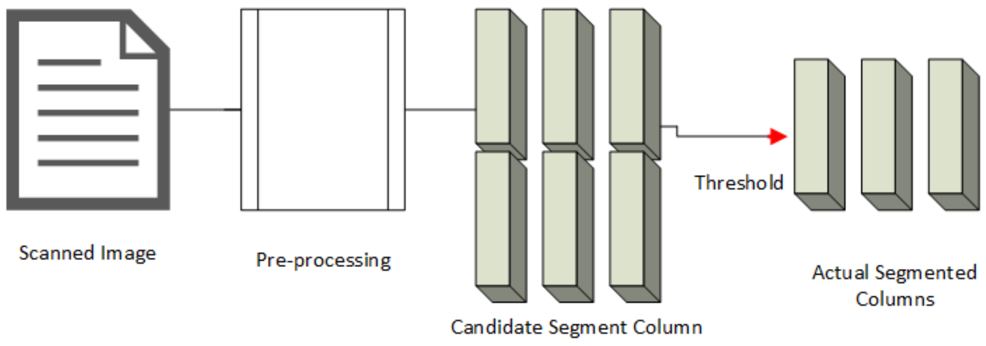


Figure 2.4: Segmentation Algorithm [3]

heuristic feature detection algorithm, first average character width of character is estimated, second, lower and upper contours are obtained to locate minima in a word then a histogram of vertical pixel density is achieved to confirm segmentation points in word, words are scanned for proper distribution of segmentation points, clusters are created for segmentation points if width of area in word is larger than average character width than it has small distribution of segmentation points.
Segmentation point is merged into more area of word segment. All segmented points are trained and tested by artificial neural network, overall word recognition was not high enough.

Amjad et al. [3] method which first performs pre-processing on image, it binarizes the image with Ostu threshold algorithm then slant correction is performed after pre-processing, sum of foreground pixels are calculated for each column and saved as separate candidate segment columns of sum 0 or 1, but it generates too many CSC by restricting it to a certain threshold actual segmented columns are achieved, then coordinates of all segmented points are detected and used for ANN training, this approach performed well for all cases except for few characters. (See Figure. 2.4) represents segmentation algorithm.

2.4.2 Chinese Language Segmentation Algorithm

Formally cursive language like Chinese, word segmentation problem solved by fixed size local windows by capturing information between interaction of adjacent tags, but through neural network they achieved better performance, proposed technique by Deng [26] generates multiple distribution of words and forwards it to long short-term memory. Another transition based neural word segmentation [27] for Chinese language which gives high accuracy by replacing hand crafted features with neural features. Another segmentation algorithm [28] which is improved version of forward maximum matching algorithm along a dictionary provided efficient results of segmentation dilemma.

2.5 Segmentation Based Approaches for Urdu Scripts

Misbah et al. [29] methodology is three phase process, initially details are collected and probabilities of ligatures and words are calculated, in second phase, for given input.
All types of words are being generated in sequence and ranking this sequences through lexicon lookup, and only few top sequences are selected, in third phase, word language model and ligature model is used, relationship between word and ligature generates bi-gram. By using trigram probabilities word identification rate is 96.10 %.

Nadir et al. [21] proposed a technique for segmentation which handles space omission problem (See Figure.(2.5)) by first preprocessing input by removing diacritics and normalizes input text, tokenization of input text produces orthographic word (OW), multiple OWs provides multiple segmentations which are forwarded to maximum matching module and ten best segmentations are selected based on minimum word heuristics. Space insertion problem (See Figure.(2.6)) first handles reduplication by single edit distance. English abbreviations are fixed by automaton. An affix list is generated to check if OWs can be combined. Lexicon is generated from corpus which identifies an Urdu compound. All segmentations are re-ranked through N-gram based ranking methods and outputs are in form of sequence of tagged words. Overall accuracy of technique is 95.8 %.

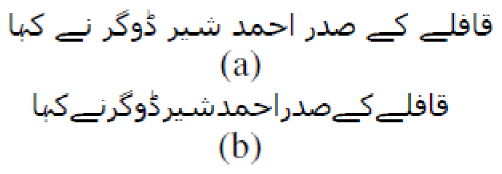
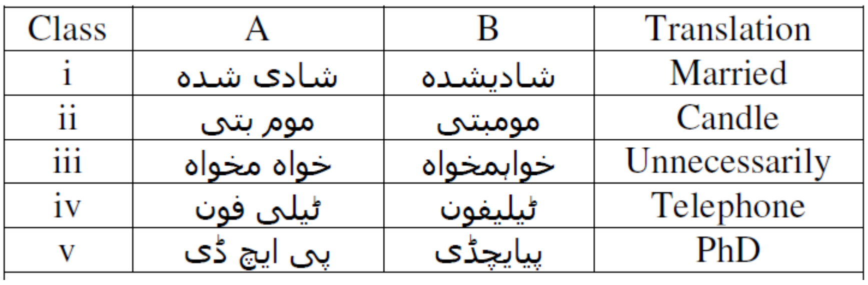

قافلے کے صدر احمد شیر ڈوگر نے کہا
(a)
قافلےکےصدراحمدشیرڈوگرنےکہا
(b)

Figure 2.5: Space Omission Problem



Class	A	B	Translation
i	شادی شدہ	شادیشدہ	Married
ii	موم بتی	مومبتی	Candle
iii	خواہ مخواہ	خواہمخواہ	Unnecessarily
iv	ٹیلی فون	ٹیلیفون	Telephone
v	پی ایچ ڈی	پیایچڈی	PhD

Figure 2.6: Space Insertion Problem

2.5.1 Pre-Processing

Gurpreet [30] introduced a hybrid approach which uses top-down method for line segmentation and bottom up for line into ligature segmentation. At first, image is broken down into horizontal lines, second is estimating accurate row height through inter peak gaps and connected component heights, then zones are labeled based on height and merged later with respect of row gaps these merged zones splits into text lines. In this case ligature might overlap, to segment ligature properly connected component technique is used with some hand-crafted rules. Accuracy is 99.02%.

Ibrar et al. [1] proposed methodology which converts an image into lines and uses horizontal projection based segmentation method with curved line split algorithm, it starts a demarcation line from middle of base line and moves from left to write if it finds text pixel it moves vertically up and down otherwise it traverse horizontally.
Ligature segmentation algorithm extracts text through connected component technique and divides it into primary and secondary components, secondary component is merged with primary on the bases of width, height and other features, accuracy of line and ligature is 99.33% & 99.80 % respectively.

2.6 Feature Learning for Arabic Script

The feature extraction stage uses a set of moment invariant descriptors which are invariants under shift, scaling, and rotation, the actual classification is done using a multilayer perceptron network with back-propagation learning [31]. This system showed high accu-

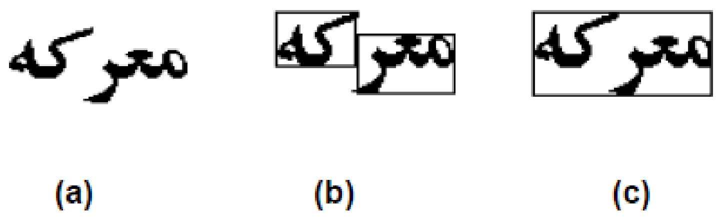


Figure 2.7: Arabic word, Connected Component and Group [4]

racy for Arabic script.

Mandana et al.[4] method starts with pre-processing the input files, global thresholding is applied to binarize the image, and connected component recognition is used where rectangular is created to group the connected regions, (See Figure. 2.7) represents connected component recognition. Skew detection and correction is used to determine angle of each group and for feature extraction contour tracing is further explained. Robust recognizer for Arabic script is a challenging task specifically in case of highly altering data such as videos or natural scenes. Defining manual features will never be able to provide as good results as deep learning which automatically detects features and moves to segmentation-free approach which is discussed briefly in up-coming sections.

2.6.1 Feature Learning through CNN

Sajjad et al. [32] achieved high accuracy through convolutional neural networks on hand written Arabic and Farsi digits. Automatic extraction of Farsi digit features performed by CNN and rejection strategies were used to find out the samples which are hard to recognize. Mohit et al. [33] applied CNN-RNN hybrid approach on natural scenes and video for recognition of arabic vocabulary and their method outperform the state-of-the-art methods. As CNN provides a segmentation free approach which solves half of the problem and manually featuring issue can be solved.

Mustapha et al. [5] compared convolutional neural network (CNN) and hand-crafted features for Arabic hand writing with hidden markov model (HMM), an analytical and character modeling approach to determine which technique retrieves best features. CNN

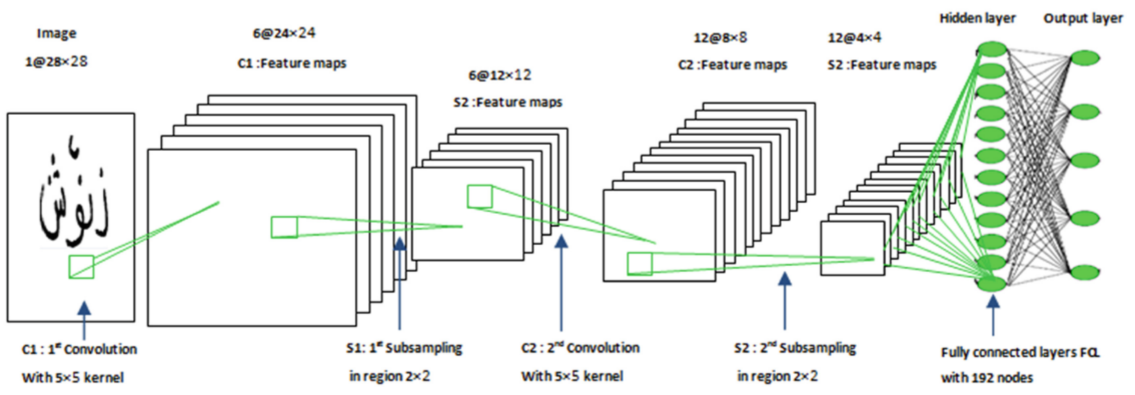


Figure 2.8: A typical CNN architecture for proposed CNN for offline Arabic handwriting recognition [5]

achieved best results but the low performance of hand-engineered features were comprised due to lack of supervision and optimization strategy was not according to the problem. (See Figure. 2.8) is CNN feature handling.

2.7 Feature Learning for Pushto Script

This paper introduced a model where feature space is based on scale invariant feature transform (SIFT), the process passes through four stages. First, scale space extreme detection, second, key point localization, third, orientation assignments and four, key point descriptor [34]. Sift descriptors are calculated for both train and test datasets and finds maximum number of key points in target image. This method is proposed for Pushto language. Scale invariant feature transform have been used for many other languages for feature learning and successfully obtained good results.

2.8 Feature Learning for Urdu Scripts

U.pal et al. [35] character recognition is performed through binary tree classifier and for feature extraction topological, contour and water reservoir methods are combined. Accuracy achieved is 97.8% for characters only.

Inam et al. [36] character detection algorithm detects character symbols through pixels, pixel value should be black or white and input images assumed are bitmap only, after character detection pixels are stored in matrix in two steps, first, left, right and top will be stored then bottom characters. Once matrix is ready it is used as training input for feed forward neural network, parameters for neural network are pre-defined after training and testing accuracy reported of individual characters are 98.3% but the scope of paper is limited to the specific font type only.

Nabeel et al. [37] approach for isolated characters recognition first analyzes the geometric properties as well as visual and hand-crafted set of features. Set of features consists of primary stroke features and secondary stroke features and then weighted linear classifier is used to perform classification.

Zaheer et al. [38] proposed a segmentation technique which measures pixel strength and detects joint where two characters are connecting and creating a word along separate words, after segmentation procedure, segmented characters are trained through feed forward neural network. Achieved accuracy is 70%.

Malik et al. [39] proposed a holistic approach for classification, feature extraction of word is based on structural and gradient features and SVM is used for classification. SVM performs better as compare to other classifiers but in segmentation based method recognition is difficult for SVM.

Ali et al. [40] proposed approach is divided into indexing and retrieval, in indexing from printed image PWs are extracted these PWs represents features, feature extraction is based on hand-engineered features, in next phase query word is compared with indexed PWs for word retrieval.

Khalil et al. [41] two databases train and test are used, eigen values and eigen vectors are calculated for images, a feature vector and a threshold value is selected to define max distance between images of both datasets, comparison is based on the identification of matched characters in train dataset.

Shazia et al. [42] sentence boundary identification technique uses feed forward neural network for the identification of sentence terminators in Urdu text files, corpus converted into bipolar descriptor array by using word-tag frequency, final descriptor array is created

Table 2.1: Types of ligatures and corresponding recognition rates [16]

Ligature	Total Ligature	Correctly Recognized	Recog. Rate
One character	117	112	96%
Two character	99	92	93%
Three character	22	20	91%
Four character	6	5	83%
Five character	4	4	100%
Diacritics	103	102	99%

Table 2.2: Notable studies on Urdu OCR with recognition rates [16]

Study	Dataset	Recognition Rate
Z.Ahmad et al.[43]	Synthatic and real-world images of Urdu	93.4%
T.Nawaz et al.[44]	Isolated characters in different font size	89%
N.Sabhour & F.Shafait. [9]	UPTI Online Arabic e-book	Urdu:99%
D.Sathi [45]	29,517	97.10%
Javed et al. [46]	1282	92%

by applying threshold to probabilistic values and then arrays are used as training set for FFNN. It can be trained on small datasets, no hand-crafted rules are needed, and it requires less storage, precision achieved up to 93.05 % and re-call 99.53 %.

Safia et al. [16] connected components were extracted and divided into primary and secondary ligatures, each ligature represents set of features where all the ligatures are clustered in groups, (See Table. 2.1) showing ligaure recognition rates these clusters are used as training data and HMM is used for each ligature. After primary ligature recognition, secondary is recognized for completion of word.(See Table.2.2) is recognition rates for all the Urdu OCR studies. Segmentation-free approaches reduces the complexity of the system as segmentation at character level is not required but, on the other hand, it increases the number of classes to be recognized which is same as the number of unique ligatures[16].

Tofik et al. [6] OCR system performed on 23204 Urdu ligatures, after feature extraction the corresponding classes obtained were 1600. ID is associated with each feature point through calculating descriptors, features are extracted based on hand-crafted rules. Ligature in digital input image is converted to editable text by matching the ligature with

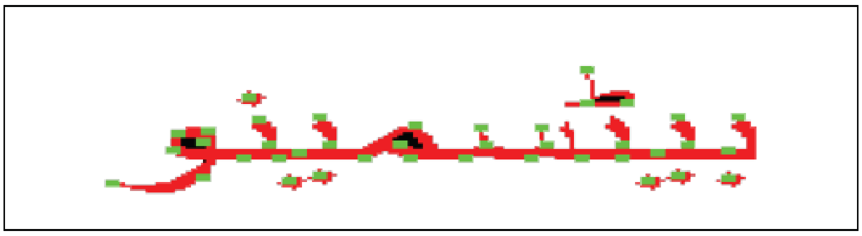


Figure 2.9: Nuqtas, Curve points and End points [6]

highest descriptor ID. (See Figure. 2.9) represents hand-crafted features.

2.8.1 Automatic Feature Learning

Saeeda et al. [47] MDLSTM technique consists of three stages based on neural networks with multi-dimensional long short-term memory by using statistical features. First stage is pre-processing and feature extraction, text is made of words and characters, text is reproduced through text line grayscale images for decoding purpose then sliding window approach is used to divide text line into characters and divides each in frame, extract features from frames and create feature vector sequence. In second stage, multi-dimensional long short-term memory is used. Input block of n*n feature vector bond with the correspond transcriptional values. Final stage CTC is used to recognize character by using conditional probability of label and input data, it selects a label of sequence with high value. The proposed approach significantly outperforms the state-of-the-art Urdu recognition system with results of training error 3.72 % and recognition accuracy is 94.97 %.

Naila et al. [2] system has dataset of 2430 ligatures, initially images are fed to the system, where global image thresholding method is applied for grayscale level of image, segmentation is performed by horizontal projection for line segmentation, sum of pixel intensities is counted for each row, for ligature segmentation vertical projection is used. Segmentation will produce image with unwanted pixels which is trimmed further. Feature extraction is performed by hand crafted rules. Once features are extracted agglomerative hierarchical clustering is used, it is performed in two levels. In level-1, 32 clusters of ligature for each feature is generated. In level-2, total number of default clusters produced by clustering algorithm have many clusters which were consists of only 1 or 2 images,

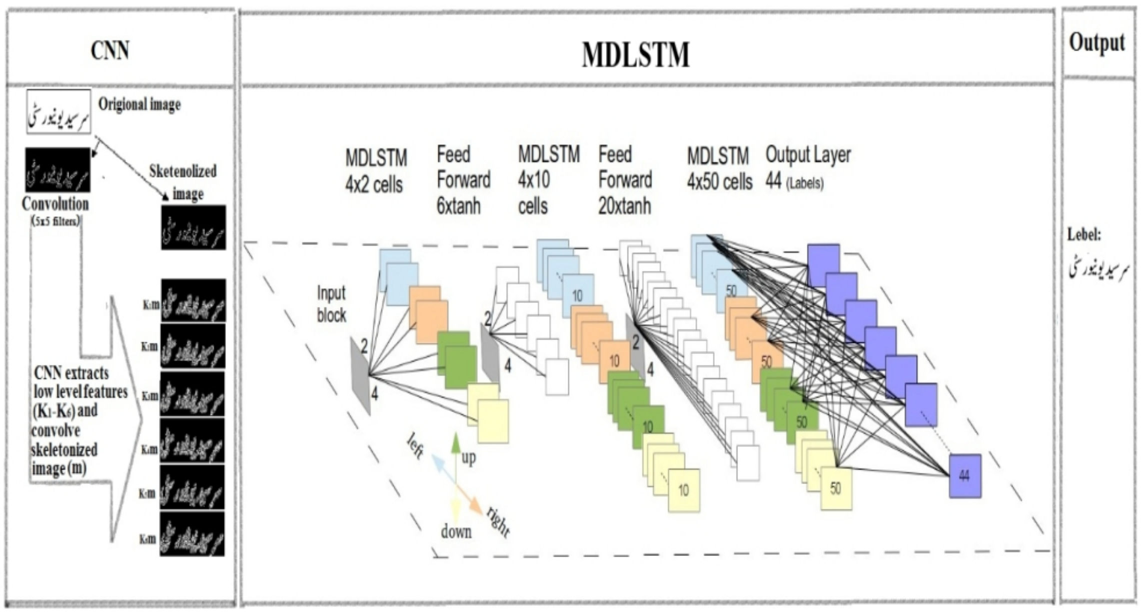


Figure 2.10: Convolution-Recursive Deep learning model [7]

to reduce the burden such clusters are put into single global cluster. Total of 417 clusters were generated, to verify 2-level clustering 4 classification techniques are used. Accuracy of decision tree, linear discrimination, naive bayes, KNN were 62%, 61%, 73%, 100% respectively.

Ibrar et al. [48] a deep learning network stacked denoising autoencoder for automatic feature extraction of ligature image pixel values used for Urdu OCR. Autoencoder is consist of feed forward neural network and non-recurrent neural network, an autoencoder is multilayer perceptron but it has equal number of inputs and outputs. SDA consists of two steps pre-training and fine tuning. First deep stack of denoising autoencoders are pre-trained on Urdu ligature images in an unsupervised manner layer by layer then this trained layer is forwarded to the MLP which fine-tunes it through back propagation and classifies it through logistic regression. Accuracies reported are 93% to 96%.

Saeeda et al. [7] hybrid approach proposed using CNN and MDLSTM, first CNN extracts lower level features from MNIST database to train the network as languages are stroke base and MNIST digit strokes have similarity with Urdu cursive script. As kernel learns features and then convolves it with Urdu text line and MDLSTM is used to classify it further. CNN extracts low level features from Urdu text line and applies six filter along

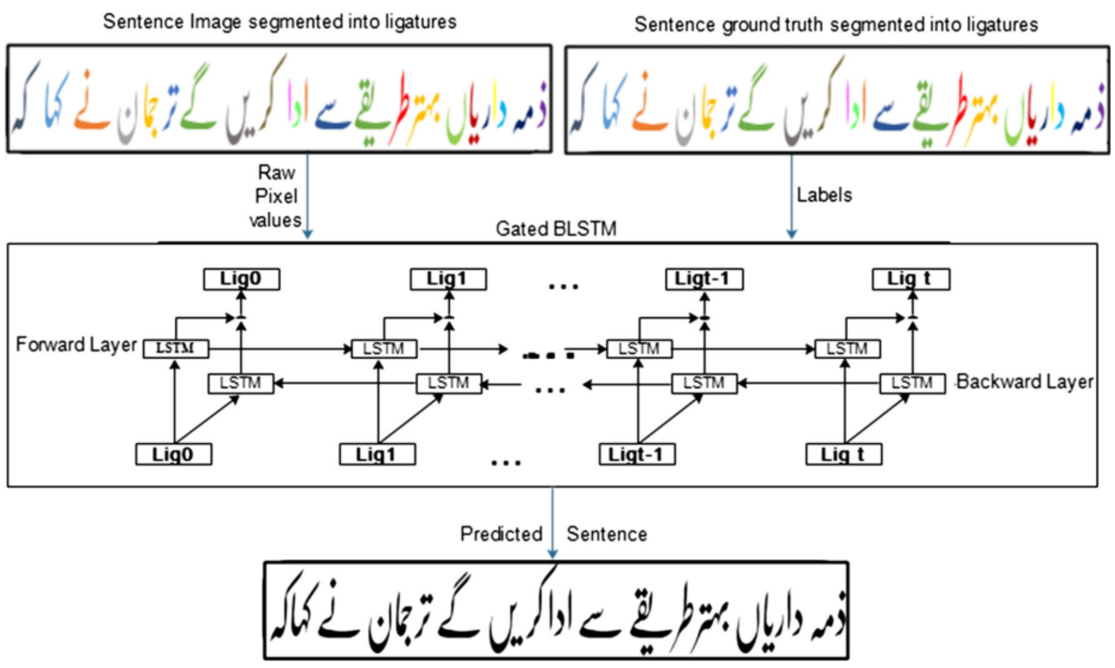


Figure 2.11: GBLSTM Network[8]

contour filter it produces contour image and filtered images which works as input for MDLSTM and uses random weights which maps features into a lower dimensional space, and performs concatenation of all vectors for CTC layer. (See Figure. 2.10) for proposed CNN approach.

In case of extracting context based information which is useful for making sentences and needs to be sementically correct ligature may not provide context information and in this case character classification might be useful, to over come such situation ibrar et al. [8] implemented a gated bidirectional long short-term memory which is based on recurrent neural network-LSTM for recognition purpose. The method extracts features from raw pixel, first model is trained on noise free version of images and tested by feeding degraded images, accuracy achieved is 96.71%. (See Figure. 2.11) represents process.

Nazlay et al. [9] developed an application, Nabocr is an OCR system specifically used for Arabic script, it is trained for both Arabic and Urdu scripts. The framework of Nabocr contains three parts, initially it trains raw Arabic script and outputs a dataset which consists of ligature in form of feature vector, after training process recognition part takes place, it takes an image as input and converts an image into text by using dataset of ligatures, third part of application is user interface as it enables user to edit recognition output. (See Figure. 2.12) is dataflow for Nabocr.

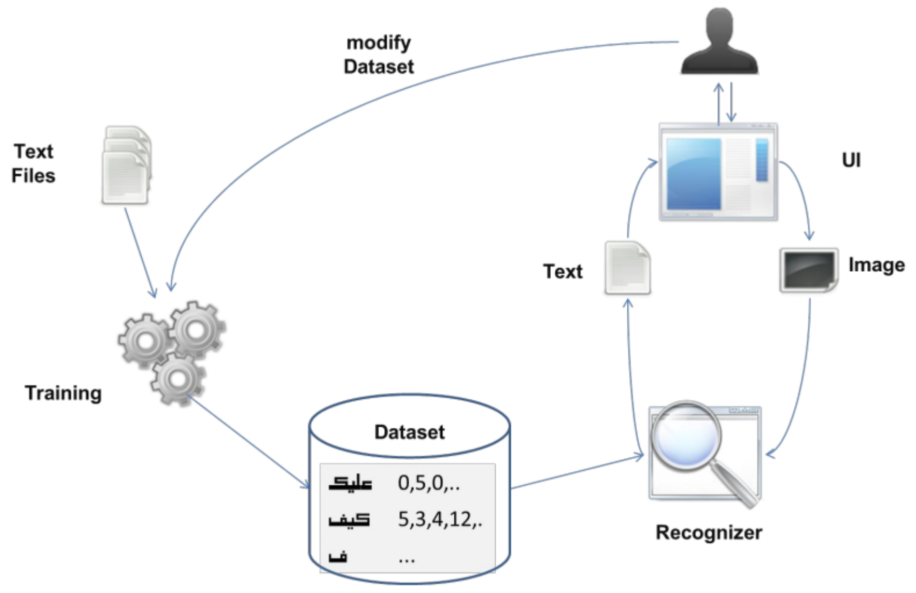


Figure 2.12: Nabocr [9]

2.9 Connected Component labeling Algorithms

Connected component method is used to separate the components by assigning a different label to each component. It is concept of computer vision where regions that are connected to each other are being labeled. It can be used for binary images and for many other image recognition tasks. Connected component can be used for many other tasks as well. This section explains different connected component algorithms which produced best results for complex images such as stroke width transformation algorithm and two-pass connected component algorithm.

2.9.1 Stroke Width Transformation

Boris et al. [49] stroke width transformation (SWT) uses an image operator to calculate stroke width of each text pixel, then each pixel is merged into letters and grouped it into words. SWT converts gray values of image into likely stroke width array and extracts text from components through measuring width variance as text in natural image which maintains same width. SWT generated best results for curved scripts as well but problem with this technique is small or thin text remain undetected.

An updated algorithm for stroke width transformation [50] is introduced which had overcome all short comings of previous one and can detect small or thin lines as well as can be used for cursive languages. It obtains a RGB image and segments the components. It can detect in any direction or scale also any font and language can be detected through this updated stroke width algorithm.

2.9.2 Two Pass Connected Component Algorithm

Kesheng et al. [51] introduced two improvements in connected component labeling algorithm, called two pass connected component labeling algorithm. In first pass accessed neighboring pixels are reduced by decision tress and in second union-find algorithm is streamlined for tracking equivalent labels.

2.10 Machine Learning

Dataset is divided into train and test datasets. The learning pipline of machine learning depends on training and testing data which refers as a supervised learning. A process is higher order function and applied on training day for learning, which will return us a hypothesis which is a cheap version of a process, in order to predict the split test data hypothesis will be used. The predicted result by hypothesis will generate an error, the difference between predicted results and expected results are known as error, cost or energy function.

2.10.1 Linear Regression

Training is based on learning features, a pixel of image represents feature, i.e. ‘x’ is our feature, ‘R’ is real numbers ‘m’ are our data points and ‘n’ is number of features, trained against labels ‘y’ or targets.

$$X \in \mathbb{R}^{mxn} \tag{2.1}$$

where,

$$y \in \mathbb{R}^{m} \tag{2.2}$$

As ‘R’ is continuous so type of learning will be regression. Relationship of ‘x’ and ‘y’ is linear means ‘x’ is some combination of ‘y’, i.e. we assume any two values ‘a’ and ‘b’ which specifically returns a deterministic system without any randomness.

$$y = a + bx \tag{2.3}$$

Hypothesis by assuming theta values:

$$h_{\theta}(x) = \theta_0 + \theta_1(x) \tag{2.4}$$

In above equation (2.4) theta-0 is our intercept in case of no value for ‘x’ it will output value for ‘y’ and theta-1 is a slope, by providing both theta values it will produce some hypothesis, if our hypothesis successfully achieved ‘y’ it means predicted value is close to the expected value and assumed theta values are correct for hypothesis. Therefore now our goal is to find the correct values of theta means fitting the parameters. In case of many features hypothesis equation (See Eq.2.5) will be transformed and ‘1’ will be padded by replacing theta-0.

$$h_{\theta}(X) = \Theta(X) \tag{2.5}$$

The overall error generated by hypothesis must be reduced, if our error is increasing and decreasing with the time, it will be unbiased and preferred rather than a systematic error which is always high or low, to balance out the error (preventing it from generating it negative value) and find out the total difference between predicted and expected result we will use square, by using square if even our difference is small now it will exponentially increase.

error is:

$$h_{\theta}(x^{(i)} - y^{(i)})^2 \tag{2.6}$$

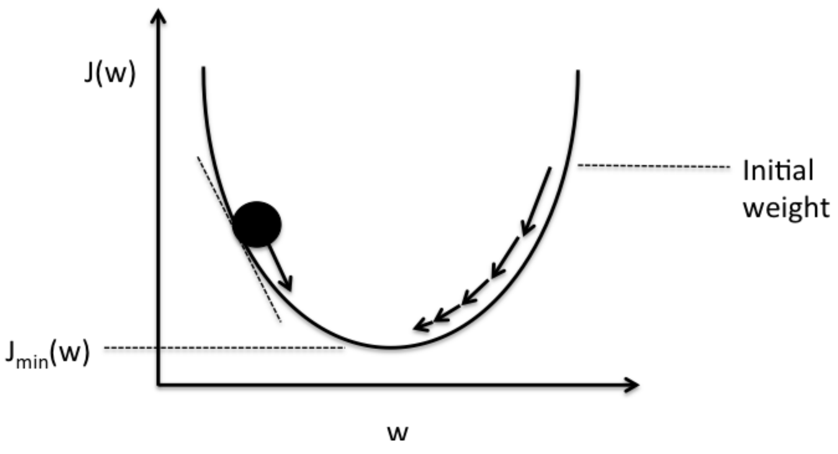


Figure 2.13: Gradient Descent [10]

by taking sum of error:

$$\sum(h_\theta(x^{(i)} - y^{(i)})^2) \tag{2.7}$$

error will keep increasing with large number of data points, for example if our dataset is too large the error will be larger i.e. we take average of error to reduce the error rate and called it as mean squared error (See Eq. (2.8)).

$$\frac{1}{2m}\sum(h_\theta(x^{(i)} - y^{(i)})^2) \tag{2.8}$$

2.10.2 Gradient Descent

Mean squared error (See Eq.2.9) is a convex function because after reaching to the lowest point you will move upwards only and will create a bowl shape structure, ‘ J ’ is our target function using theta-0 as a constant and different values of theta-1 (See Figure. 2.13).

$$J(\theta_0, \theta_1) = \frac{1}{2m}\sum(h_\theta(x^{(i)} - y^{(i)})^2) \tag{2.9}$$

If we take derivative of given error it will give us a slope which is combination of direction and length, by following the direction it will generate minimum error. Following equation (See Eq.2.10) updates theta values, in case of higher theta value divergence will achieve and for lower values it will take more time to converge which can cause our algorithm to stuck in infinity, to overcome this situation we will use alpha a learning rate to control convergence and divergence but to set learning rate needs careful consideration. Gradient decent uses batch gradient which computes error on all data points at each iteration. update theta:

$$\theta_j := \theta_j - \alpha \frac{\delta}{\delta \theta_j} J(\theta_0, \theta_1) \tag{2.10}$$

by taking partial derivative of theta (See Eq.(2.11)) so that weights can be updated in some cases this weight can do very small value and can cause vanishing gradient problem. This problem can be solved through Resnet CNN this technique is explained in further sections.

$$\frac{\delta}{\delta \theta_j} J(\theta_0, \theta_1) = \frac{1}{m} \sum (h_\theta (x^{(i)} - y^{(i)}) x^{(i)}_j) \tag{2.11}$$

2.10.3 Stochastic Gradient Descent

In case of larger number of data points batch gradient will not work and can cause bottleneck therefore we replaced sum by randomly selecting few data points called mini-batch gradient (See Eq.2.12), the randomly selected samples will not be repeated more than once.

$$\frac{\delta}{\delta \theta_j} J(\theta_0, \theta_1) = \frac{1}{m} \sum (h_\theta (x^{(i)} - y^{(i)}) x^{(i)}_j) \tag{2.12}$$

The effect of a specific feature will be much higher with respect of the scale of a feature. Therefore we will normalize features (See Eq.2.13) between range of ‘0’ and ‘1’ and effect of all the features will be equivalent which is called as feature scaling.

$$h(x) = \frac{x - x}{max(x) - min(x)} \tag{2.13}$$

2.10.4 Classification

In classification output can be ‘0’ or ‘1’ same as ‘yes’ or ‘no’ but it can generate bad error in case of extreme cases and may not occur between ‘0’ and ‘1’.

$$0 <= h_\theta(x) <= 1 \tag{2.14}$$

In such case where we want to normalize output between ‘0’ and ‘1’ is called sigmoid function (See Eq.2.15) or logistic regression, eventually no matter how bad error is output will be between ‘0’ and ‘1’.

$$h_\theta(x) = g(\theta^T x) \tag{2.15}$$

where,

$$g(z) = \frac{1}{1+e^{-z}} \tag{2.16}$$

2.10.4.1 Cost Function

A piece-wise function to calculate error or cost function (See Eq.2.17) for a logistic regression, along a squishing function, piece-wise function can also use if-else conditions but derivative of if-else can be a complex procedure.

$$J(\theta) = \frac{1}{m}\sum Cost(h_\theta(x^{(i)}, y^{(i)})) \tag{2.17}$$

where,

$$Cost(h_\theta(x^{(i)}, y^{(i)})) = \begin{cases} -log(h_\theta(x)) & \text{if } y = 1 \\ -log(1 - h_\theta(x)) & \text{if } y = 1 \end{cases} \tag{2.18}$$

which outputs,

$$J(\theta) = -\frac{1}{m}[\sum y^{(i)} log h_\theta(x^{(i)} + (1 - y^{(i)}))log(1 - h_\theta(x^{(i)}))] \tag{2.19}$$

2.10.4.2 Regularization

Linear model for a quadratic ground truth can cause high variance where training error is much smaller than the testing error, adding more data may not lead to lower a variance if a problem exists in underlying structure, in a quadratic ground truth order of polynomial can be larger and in turn larger values of theta will effect higher which consequently produces high variance, the technique known as regularization (See Eq.2.20) is used as a solution here, by introducing a lambda which is regularization parameter in a cost function will decrease the effect of higher theta value up to certain level and the effect of particular features will be muted, and higher lambda value will decrease the test error accordingly.

$$J(\theta) = \frac{1}{m}\sum Cost(h_\theta(x^{(i)}, y^{(i)}) + \lambda \sum(\theta_i)^2) \tag{2.20}$$

2.11 Artificial Neural Network and Deep Learning

Machine learning explains the core concept of artificial neural network and how hidden layers are introduced in this and their functionality. This sections contains the detail view of artificial neural network and leads to the convolutional neural network for the proposed method.

2.11.1 Biological Motivation

Human brain is a structure of massive number of neurons, which acquires an intelligent behavior and most robust learning system. The massive parallelism of neurons allows an efficient computational system along dispersed nature of neuron helps to make it robust. Neural networks [52] are inspired from actual human brain. Real neuron cell structure is consists of cell body, axon, dendrites and synaptic terminals, neurons communicate with each other through neurotransmitter and spikes in cell body fires action potential. Spikes which passes through the axon makes synaptic terminal to release neurotransmitter, a chemical diffusion occurs and spreads across all the dendrites, if neurotransmitter is above a certain threshold an action potential is fired.

2.11.2 Perceptron

In 1943 Warren McCullah and Walter Pitts [53] introduced a simple artificial neural network called perceptron (See Eq.2.14), ‘x’ represents the input and ‘w’ represents the weight or strength of each input, sum of input ‘x’ and weight ‘w’ is called activation. Inputs and assigned weights represents the strength of a connection if sum of both the input and their weights are above a certain threshold it fires for instance and it will fire if output is above 0.5 only, same as biological one. Explained in mathematical terms (See

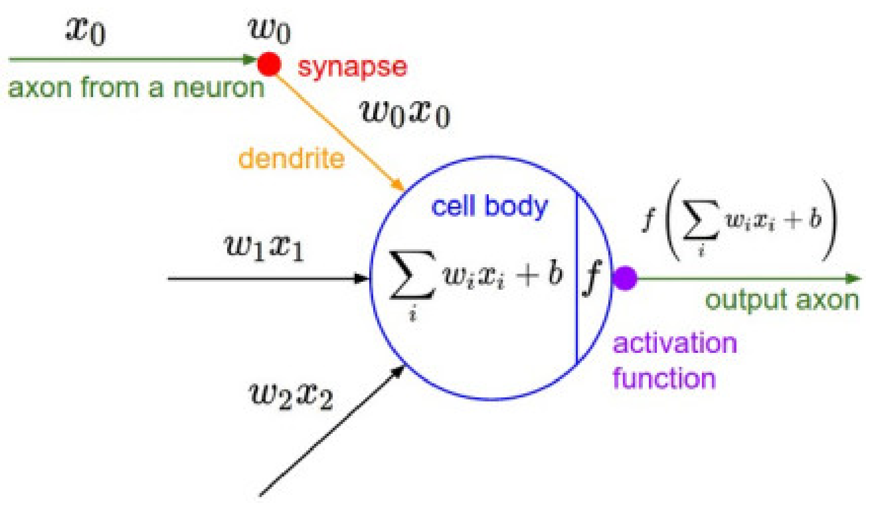


Figure 2.14: Perceptron [11]

Eq. 2.21) and (See Eq. 2.22).

$$O(x_1, ...x_i) = \begin{cases} 1 & \text{if } w_0 + w_1 x_1 + ...w_n x_n > 0 \\ -1 & \text{otherwise} \end{cases} \tag{2.21}$$

Considering input and weights as a vector:

$$O(X) = \begin{cases} 1 & \text{if } W.X > 0 \\ -1 & \text{otherwise} \end{cases} \tag{2.22}$$

The activation function is binary here, will output in ‘1’ or ‘0’ but to obtain continuous result which are more flexible it changed to sigmoid function (See Formula. 2.23). It called as squashing function as it produces output between ‘0’ and ‘1’. Other activation functions Tanh or Linearlog can also be used.

$$O = \frac{1}{1 + e^{-S}} \tag{2.23}$$

2.11.3 Back Propagation Algorithm:

Back propagation is training and learning algorithm, it learns input at training stage by updating weights and produces required output when a particular input is given. The main

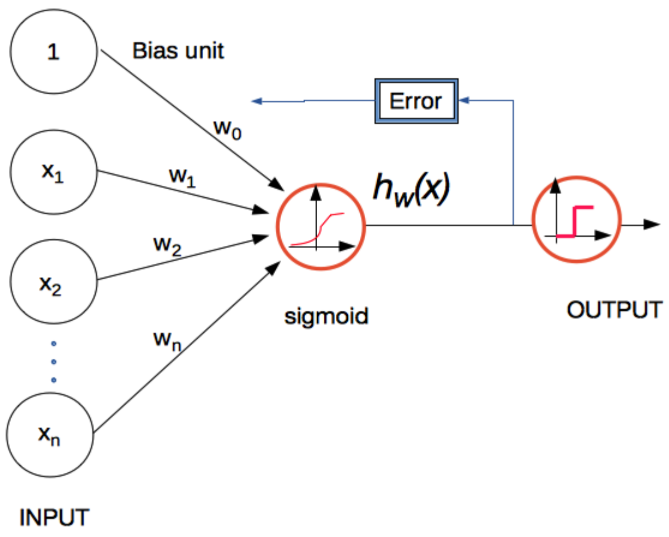


Figure 2.15: Back Propagation Algorithm [12]

goal of neural network is to find the suitable linear function which fits the data but to add more variety in function more layers are need to be added for that we use back propagation algorithm. Network is trained on the examples, more examples lead to better training and target output can be easily recognized. Back propagation produced some ideal result in pattern recognition. Initially weights are assigned randomly, in first forward pass output is calculated through inputs and weights, if output is not same as actual target class it will be called error (See Eq.2.25), to minimize the error we will update weights (See Eq.2.24) at each iteration until error is reduced to the level where difference between output and actual target is low. Weights are updated iteratively until convergence is achieved and data is linearly separable.

$$w_i = w_i + \delta w_i \tag{2.24}$$

where,

$$\delta w_i = \alpha(target - output)x_i \tag{2.25}$$

Alpha is learning rate a smaller constant value (0.001)

Calculate error for each neuron and sum all errors to obtain total error of a network through mean squared error (See Formula. 2.26).

$$E[W] = \frac{1}{2}\sum_{d\varepsilon D}(t_d, o_d)^2 \tag{2.26}$$

2.11.4 Supervised Learning

There are two types of learning one is un-supervised learning which is based on clustering where target outputs are unknown while other is supervised learning where target classes are known and classification can be performed through various algorithms. This section explains the important terms with respect of artificial neural network architecture in supervised learning.

2.11.4.1 Activation Function

It calculates sum of input and their weights for activation, the purpose behind using activation function is that output neuron should consider this neuron as activated or not. Threshold activation function imposes a certain barrier if value is above the threshold, neuron is fired, but in case of multiple neurons, above the threshold value binary activation may not work. Sigmoid function is another activation function which results between '0' and '1' but it causes vanishing gradient problem and slow convergence, hyperbolic tangent output range between '1' and '-1', now rectified linear unit (ReLU) became more popular, it is simple, efficient and rectifies vanishing gradient problem. SoftMax is usually used at the output layer as it returns results based on the highest probabilities.

2.11.4.2 Error Measure

To minimize the error of network or also called as objective function weights are adjusted through out training. During training when first output is generated it gets compared with the actual output the difference of both is called error, to reduce the error throughout training is main goal. Mostly mean squared error is used or depending on the categorical data cross-entropy used for error measurement.

2.11.4.3 Training Protocols

Stochastic gradient selects a random input value for training and updates all the weights after error, while batch training processes all input data and calculates overall error (See

Eq.2.26). Mini-batch selects sample or subset 'M' (See Eq.2.27) of dataset for training and cumulative weights are calculated.

$$M \subseteq \{1, ...N\} \tag{2.27}$$

where subset error is:

$$E_M(w) := \sum_{n \varepsilon M} E_n(w) \tag{2.28}$$

2.11.4.4 Optimization Technique

Gradient descent is an optimization technique it takes first derivative (See Eq. 2.29) of error and finds the minimal error by moving to the steepest direction. It converges with minimum error and uses small learning rate, in case of noisy data and when separation is not possible it generates best possible separation.

$$\nabla E[W] = \left[\frac{\delta E}{\delta w_0}, \frac{\delta E}{\delta w_1}, ... \frac{\delta E}{\delta w_n} \right] \tag{2.29}$$

Training weight updates:

$$\Delta W = -\alpha \nabla E[W] \tag{2.30}$$

I.e.:

$$\Delta w_i = -\alpha \frac{\delta E}{\delta w_i} \tag{2.31}$$

2.11.5 Regularization

Overfitting occurs when model works fine on training data but generates poor result on testing data, overfitting can be prevented by adding more data or introducing diversity in data, data augmentation is another technique which acquires many properties such as it flips, rotates, zoom and allows us to add augmented images. Dropout [13] is a regularization technique which randomly ignores subset of layer during training, it will help model to generalize data it has not seen before, it adds a constant value into loss function and penalizes the complexity of model which aims to minimize loss of a network.

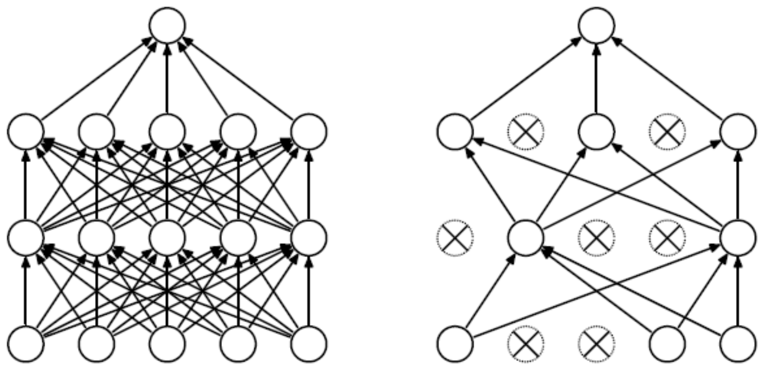


Figure 2.16: (a) Neural Network before adding dropout (b) Neural Network after dropout [13]

The problem that neuron weights can get very specific based on the context and model will get good results only for the training data. To avoid this generalization dropout is used and due to this neural network will be less specific and model will be robust enough to obtain good results for unseen data.

2.12 Deep Learning

Data can vary basis on internal patterns and structures, to learn such data representation, model is used to create multiple levels of abstraction by forming different levels of layers and extracting data representation at each layer. Deep learning made some phenomenal improvements in the field of speech-recognition, object detection and image processing such as OCR of hand-written documents, also numerous other domains as bioinformatics, robotics, identifying tumors in medical diagnosis, drug discovery and security purpose. Pattern- recognition requires feature engineering which can be detected by a classifier due to the need of conventional machine learning methods which processes data in raw form, careful feature-engineering expertise must be considered first.

Deep learning [54] employs data representation methods, initially an image which is an array of pixels or raw input is fed to the classifier, at first layer low level representation is extracted such as edges, next layer extract motifs another layer detect parts of object and at final layer object is recognized (See Figure.2.17). The main feature behind representation

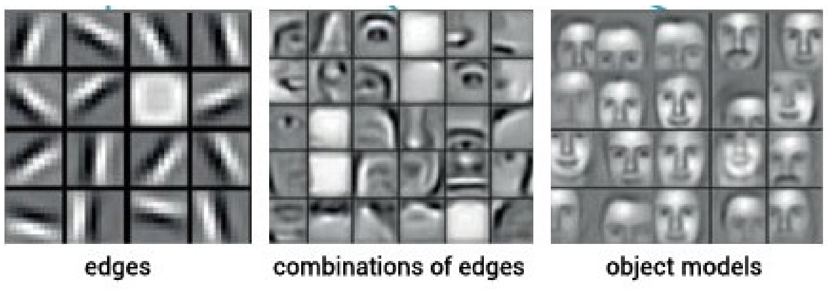


Figure 2.17: Feature Extraction [14]

learning is automatic feature detection, no human expertise are required. Deep learning automatically learns features and recognizes object by adjusting the internal parameters called weights and reduces the error. Weight vectors are adjusted through gradient vector, miner change in a weight increases or decreases the error. In practical many optimization techniques are used to compute errors and adjusts the weights. Repeating the process on training set, test is used to measures the performance of system, which produces result based on the ability of system for the inputs unknown during the training. Deep learning can also be used for unsupervised learning.

2.13 Convolutional Neural Network

Convolutional neural network is a deep learning architecture based on multilayer perceptron and owns characteristics of shared weight, sparse interaction and equivalent representation. CNN is based on the basic concepts of machine learning such as gradient decent. Convolutional neural networks successfully achieved high performance in the field of image processing due to automatic feature learning. The design of ConvNet depends on the kernel or filter, the data points in filter are much smaller than input size. ConvNet returns higher value output when filter value matches to that particular portion of an image, filter slides over the input and calculates similarity. Fully connected neural network is densely connected neural network and convolutional neural network is sparsely connected network which makes it simpler than fully connected neural network, it accepts lower number of connections which reduces number of interaction between input and output and reduces the capacity while providing better training efficiency.

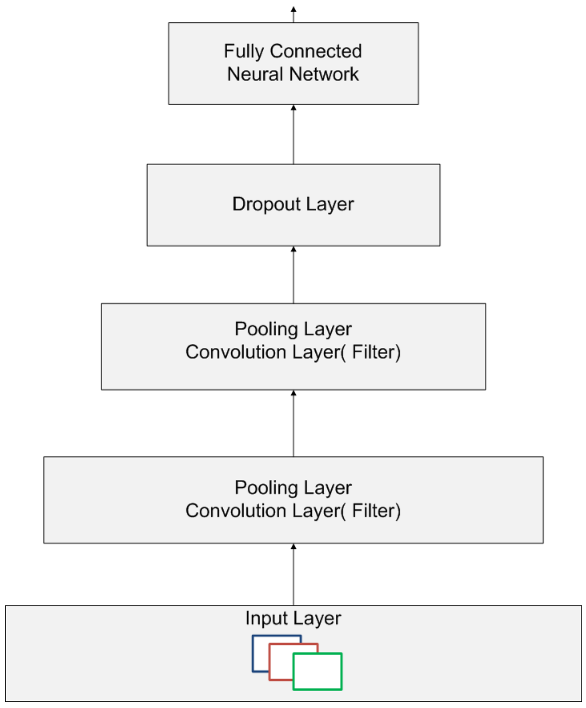

Fully Connected Neural Network
Dropout Layer
Pooling Layer Convolution Layer(Filter)
Pooling Layer Convolution Layer(Filter)
Input Layer

Figure 2.18: ConvNet Layers

2.13.1 ConvNets Architecture

ConvNet works on four key ideas [54] layers, pooling, weights and local connections. The structure of ConvNet is composed of stages (See Figure. 2.18). Initial stages consists of convolutional layers and maxpool layers along weights called filter, different number of filters passes over the single image and creates feature map, sum of the weights is passed through the activation function such as ReLU, or LeakyReLU to squash the input value into a range. Convolutional layer detects conjucted features of previous layer and semantically similar features are merged by pooling layer, max pool layer fetches a higher or maximum bit of a patch and reduces the dimensions. Back propagation end-to-end supervised learning algorithm is used to train all the weights of every filter in convolutional neural network. Convolutional neural network takes an image as a tensor, image dimension is its width and height which represents number of pixels and depth referred as channels. A patch of an image is passes through the filter smaller than the image size, the dot product of both filter and a patch outputs a value which represents a match of a pattern. Strides are the steps which filter takes while traversing the image, larger strides will compute faster and generate small activation map as compared to smaller strides.

Convolutional layer takes a dot product of input and weight and is a 2D process, weights are shared. It will accept input as (width, height, dimension). This layer creates a kernal which is convolve with an input layer and generates a tensor. As a first layer of model input-shape keyword is used for image.

$$W_1 * H_1 * D_1 \tag{2.32}$$

Hyperparameters of CNN layer:

Number of filters: K

Size of filter: F

Strides: S

Padding: P

and produces output:

$$W_2 * H_2 * D_2 \tag{2.33}$$

where:

$$W_2 = \frac{(W_1 - F + 2P)}{S} + 1 \tag{2.34}$$

$$H_2 = \frac{(H_1 - F + 2P)}{S} + 1 \tag{2.35}$$

$$D_2 = K \tag{2.36}$$

Activation layer performance of activation function varies but non-linearity in network depends on activation function, sigmoid causes vanishing gradient problem but this can be handled by LeakyReLU. Other activations are also available such as, Randomized Leaky ReLU, SReLU, Exponantial Linear Units (ELU), Hyperbolic Tangent, Softmax.

Pooling layer function is always applied parallel to the convolutional layer, it operates on a part of image not an entire image and will produce output accordingly with reduced dimensions, pooling function normally uses max but other variables are also available such as average pooling (See Figure 2.19). It takes average or max of each feature map instead of taking as fully connected layer. Through global average pooling keyword over-

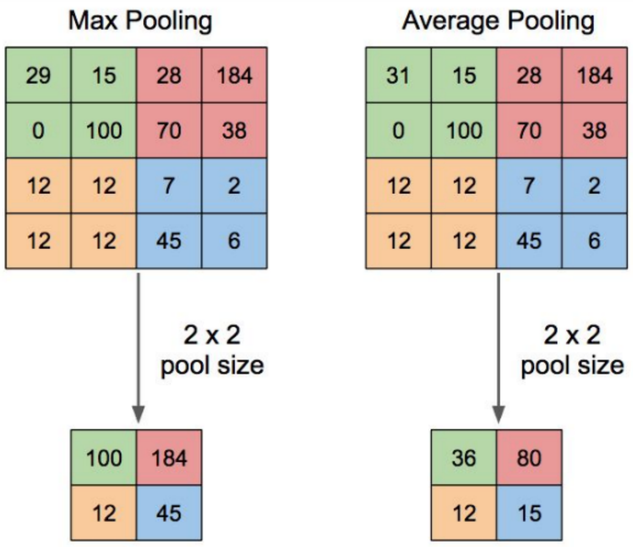


Figure 2.19: Max Pooling

fitting can be avoided because optimization of parameter is dropped.

$$W_1 * H_1 * D_1 \tag{2.37}$$

Size of filter: F

Strides: S

and outputs:

$$W_2 * H_2 * D_2 \tag{2.38}$$

where:

$$W_2 = \frac{(W_1 - F)}{S} + 1 \tag{2.39}$$

$$H_2 = \frac{(H_1 - F)}{S} + 1 \tag{2.40}$$

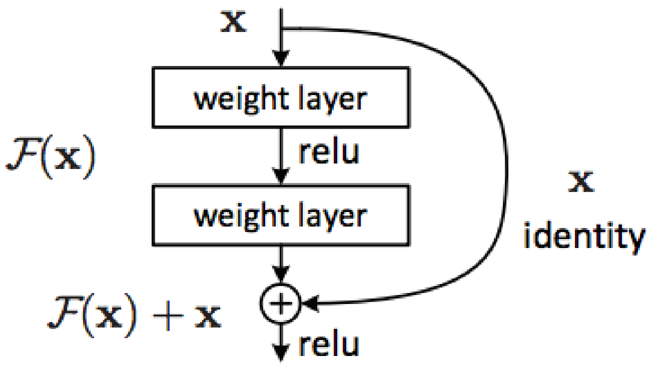
$$D_2 = D_1 \tag{2.41}$$

Fully connected layer final phase of learning, every node is connected to every output class and based on the previously computed features and maps them to the output.

SoftMax this activation function is used after fully connected layer and predicts the probability of each class.

2.13.2 Residual Nets

Deep residual learning framework [15] is simpler formation of deep neural networks, layers are being reformulated with reference of input layers. It handles the degrading problem caused by deep layer network where eventually accuracy degrades after getting saturated. Residuals solves the degradation problem, faster training then stacked deep network and higher accuracy due to lower number of parameters. Hundred layers of deep network represents complex function it helps in understanding many features at abstract level, however, it does not always helpful with causing vanishing gradient problem. Resnet adds shortcut to mainstream stacked network and allows one of the block to learn an identity function and skip connection (See Figure.2.20) helps with vanishing gradients.



x
weight layer
$\mathcal{F}(\mathbf{x})$
relu
weight layer
x
identity
$\mathcal{F}(\mathbf{x}) + \mathbf{x}$
relu

Figure 2.20: Resiudal Nets [15]

Chapter 3

Proposed Method

This section explains all the major parts of our proposed approach in detail. Started from data generation procedure and how it is further divided into component based identification. Through figures proposed method and dataset generation is well explained. Architectur of Convolutional neural network model is also given.

3.1 Dataset Generation

As per previous approaches printed Urdu text recognition was based on hand-crafted features of character and ligature but development in pattern recognition technology transformed the main-stream hand-crafted feature ideology to the automatic feature learning. Our system is entirely based on component-based identification. State-of-the-art segmentation methods often lead to the lowest recognition rates since Urdu language has problems of complex morphological structure of defining a word.

3.1.1 Component Based Identification

Feature learning needs a very large dataset to be trained initially, dataset creation is mainly depends on various phases. The major setback in Urdu research is resource scarcity for training purpose. The 83000 ligature dataset is the largest available dataset of Urdu OCR [55]. In our proposed method (See Figure.3.1) initially Naskh category fonts were se-

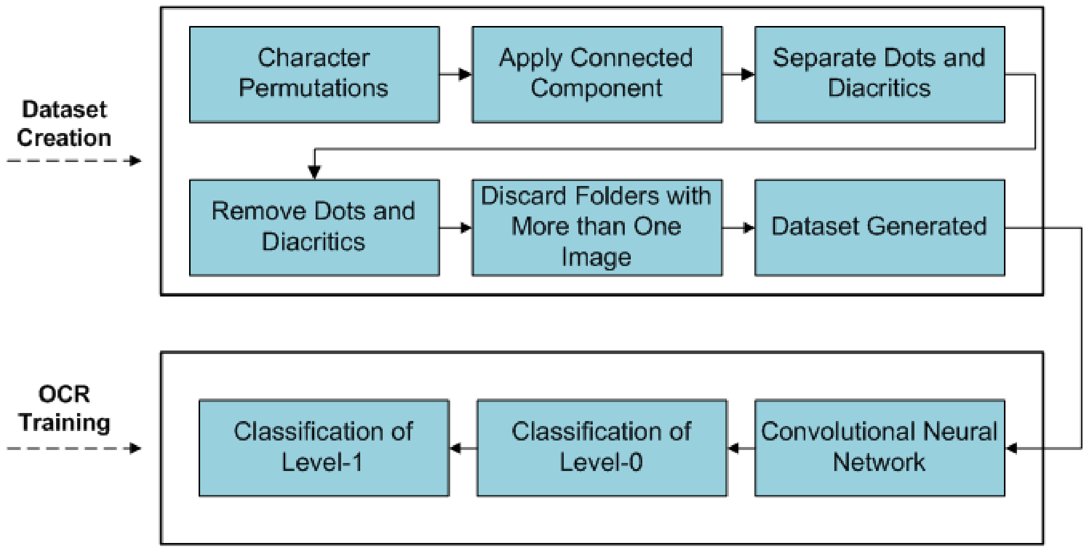

Dataset Creation
Character Permutations
Apply Connected Component
Separate Dots and Diacritics
Remove Dots and Diacritics
Discard Folders with More than One Image
Dataset Generated
OCR Training
Classification of Level-1
Classification of Level-0
Convolutional Neural Network

Figure 3.1: Block Diagram for Recognition System

lected and permutations (See Formula.3.1) which arranges all the characters in some order used for creation of component, in our case upto three character permutations were generated. The motivation behind using three character permutation is that the longest word in Urdu consists of probably seven characters while most of the Urdu words are combination of two or three characters only.

$$^{n}P_{k} = \frac{n!}{(n-k)!} \tag{3.1}$$

As a result character image of 100*100 is created, first permutation, generates images for 38 characters, second permutation creates 306 images, combination of two characters, and third permutation creates 4, 896 images through joining three characters. Number of characters can be increased depending on the need, our dataset consists of three character permutations but to train deep neural netwrok on large dataset we included 15 more fonts which makes dataset of approximately 22, 000 Urdu text images.

Connected component (See Figure.3.2) is a methodology where a set of pixels which relates to each other by means of neighborhood assigned with same class, a connected component algorithm, labels all the connected pixels in an image with a unique label if it belongs to the same component. Various connected-component techniques are part of

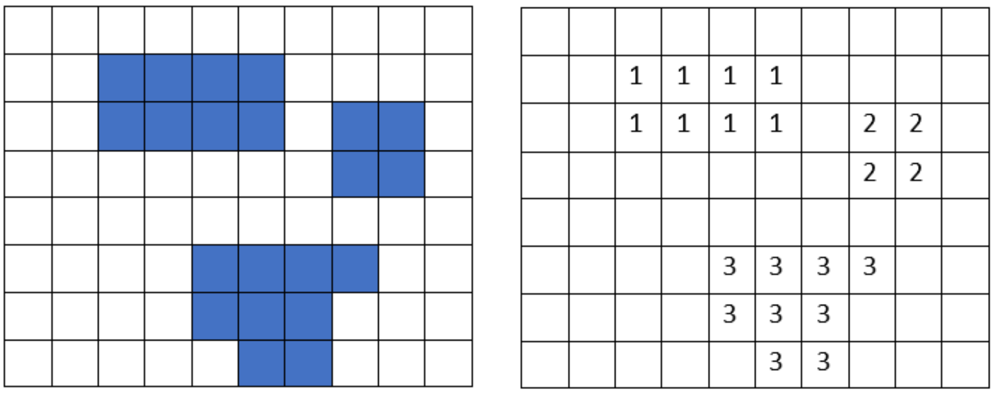


Figure 3.2: Connected Component

pattern matching and recognition and have produced some tremendous results such as, stroke-width transformation algorithm [15] and two pass connected component algorithm [51] are improved versions of connected component labeling algorithm.

Connected-component separates all those characters, dot and diacritics which are not joined, it separates dots and diacritics from words and discards them (See Figure.3.3), after removing dots and diacritics from all the permutations, characters which were not joined were discarded as well, such as, at second and third permutations many characters can not join due to the context sensitivity and will be discarded, for 38 characters dataset, after removing dots and diacritics common characters were removed as well and dataset is left with 19 characters only.

Finally dataset is consists of join characters or components only, after discarding unnecessary images our dataset is reduced to 22, 000 images for three character permutations, to increase the dataset image augmentation process is also used in architecture for better learning purpose.

3.2 Convolutional Neural Network Architecture

ConvNets are form of neural networks that have learnable weights with multiple hidden layers and properties encoded in architecture which reduces number of parameters of network. ConvNets are built upon sequence of layers, Convolutional Layer, Pooling layer

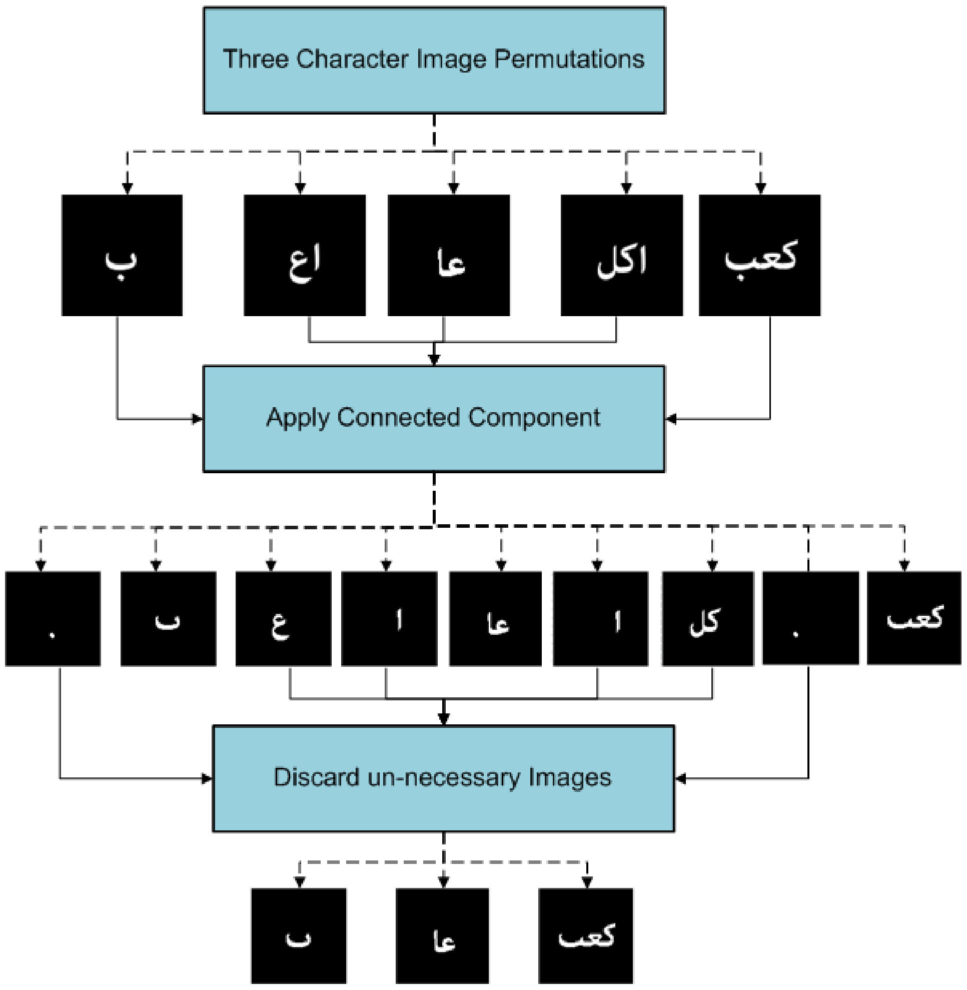

Three Character Image Permutations
ب
اع
عا
اكل
كعب
Apply Connected Component
.
ب
ع
ا
عا
ا
كل
.
كعب
Discard un-necessary Images
ب
عا
كعب

Figure 3.3: Dataset Creation

and Fully-connected layer.

Input Layer: It represents our image dimensions and channels, width and height tells us about number of pixels an image can have.

Conv2D: It acquires pixel value of images as a matrices such as, [100,100,1] width is 100, height is 100, and channel of image which in our case is 1. It also consists of number of filters and filter size which computes a dot product of input under region of filter size. Many layers of Conv2D with various size and filters are part of our model.

Activation Layer: ReLU, LeakyReLU and SReLU are activation functions to perform thresholding while SoftMax is used at the end layer to produce results with high probability.

Maxpool Layer: Pool layer down samples and takes max of each feature map which results in form of higher bit information only and discards rest of the values, such as, [100,100,32] will be [50,50,32], resultant activation map comprises higher information referred as features with strong correlation. It benefits us by decreasing the time processing needs and storage it occupies. Different pooling layers GlobaAveragePooling, AveragePooling can be used.

Dense Layer: It is fully-connected layer results in class score such as, among the 19 classes of our dataset will be separately connected to each node.

Dropout Layer: Dropout is regularization approach which adds penalty to prevent from learning interdependent weight.

Optimizers: We used RMSprop for updating weights with learning rate=0.0001.

Class Weights: Level-0 contains three number of classes and number of image for all the classes are not similar which makes our training data imbalance. To perform training for imbalance dataset class weight parameter available in keras is used. Class one assigned with highest weight as compared to other two classes.

Image Augmentation: ImageDataGenerator is capable of producing augmented images for better learning, it will create augmented images with various properties such as, zoom, rotation at different directions and flip in no time, although memory overhead reduces but it increases training time.

ResidualNets: Deeper network can cause degrading accuracy, which is not due to overfitting, by residual mapping it improves performance more than original stacked layers.

3.3 Hierarchical neural network

Hierarchical neural network [20] is multilevel neural network forms a tree structured architecture and can be used to perform hierarchical based classification for pattern recognition such as, character classification or ligature classification. Our proposed model implements a hierarchical neural architecture (See Figure.3.4) which trains input pixels and outputs a component based on the number of characters, we separately trained four neural networks at each level and joined them later. At level-0 datasets of all three characters of three classes are trained which consequently identifies the class, component belongs to, i.e. degree one represents component comprises one character, degree two component is combination of two characters and same demonstrates for degree three. The key focus of our paper is to classify the component based on characters. Following the hierarchy level-1 intends to show the component predicted by previous level. The output of level-0 is input of level-1, if a component identified as a class three at level-0 it will be push to the degree-3 where further classification will take place. Results achieved for all the datasets indicates effectiveness of our proposed scheme.

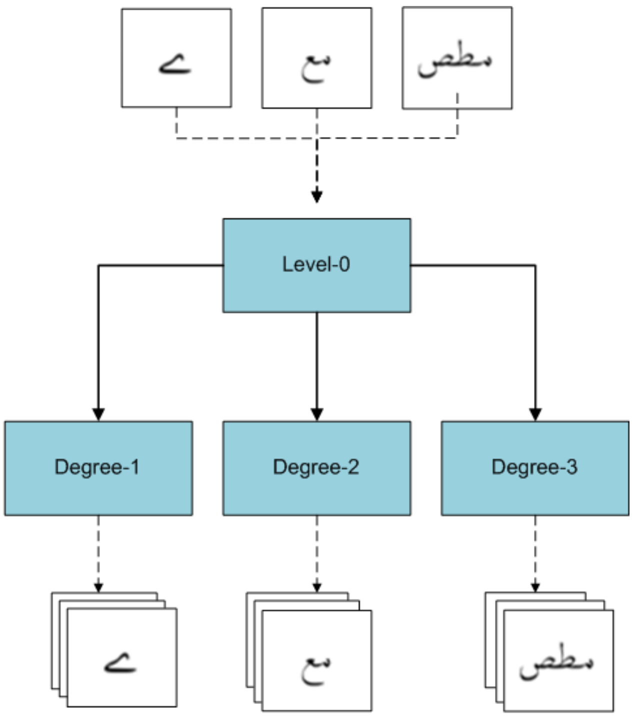


Figure 3.4: Hierarchical Neural Networks

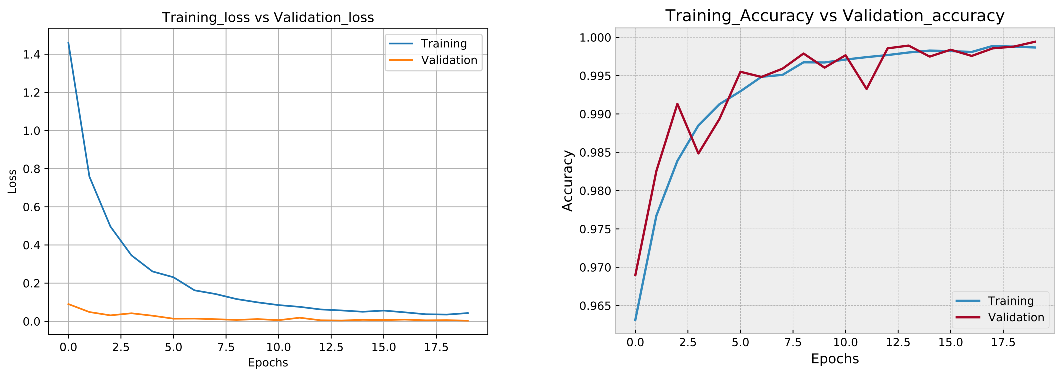
Chapter 4

Results

Deep convolutional neural network is implemented for the component recognition, keras is high-level API which is user friendly, fast and can be used on CPU and GPU both, CNN keras are sequential layers where each layer extracts different features, two to four layers of CNN in keras is created for every level and degree as previously discussed we have total four datasets divided into two levels and forms a hierarchical neural network, level-0 classifies component class and level-1 classifies component itself dataset is trained on almost 20, 000 to 66, 000 images by adding different text fonts, along sitting of hyper parameters such as kernal_size, batch_size, pool_size with different values until our model achieved high results, dropout is being added to avoid overfitting, categorical_crossentropy is used as our target labels are more than two. Basic evaluation techniques, accuracy, precision, recall and F-1 score is used to report the final results.

4.1 Level-0

Implementation of component recognition is employed by two layered CNN architecture (See Table.4.5) which is used for feature extraction of 18, 900 Urdu text ligature images of 100 * 100 as recognition of component is divided into two levels and at level-0 our goal is to classify component based on character it comprises. (See Figure:4.1(a)) is loss which illustrates error rates of training and validation datasets, error rate is decreasing with increasing number of epochs, number of epochs are 20, training error rate is 0.0013% while validation error rate is 0.0003%. Accuracy achieved (See Figure:4.1(b)) for training dataset is 0.9998% and for validation its 0.99% as dataset is imbalance in this scenario for which keras, class_weight library is used to assign weights to the classes and besides accuracy, precision, recall and F-1 score is reported as well which is 0.99% in this case. This is the perfect loss of trained convolutional neural network due to large dataset.



[a] Loss [b] Accuracy

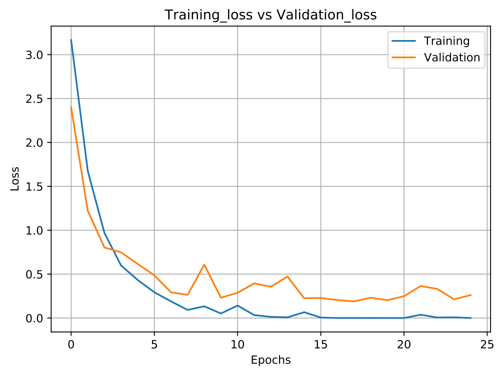
Figure 4.1: Level-0

4.2 Level-1

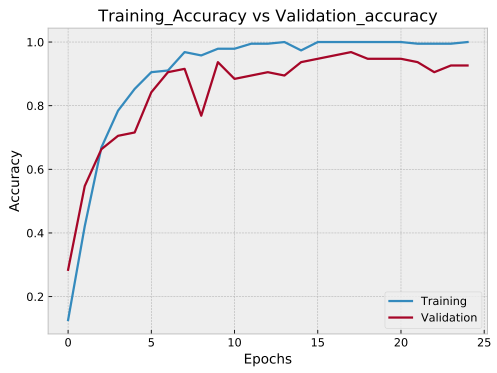
As we have implemented hierarchical neural network for component recognition where after level-0, at level-1 component recognition takes place. At level-0 obtained results are 0.99 % for recognition of category of component whether component belongs to degree-1, degree-2 or degree-3. Results of every character degree are explained in this section.

4.2.1 Degree-1

Level-1 is part of hierarchical neural network that we implemented above, after classifying character class at level-0, level-1 will recognize the character or component depending on the degree of character, for degree-1 dataset is consists of 38 characters initially but discarding un-necessary components it is left with 19 classes of characters with dataset of 285 images which is increased by adding 15 more fonts for better learning purpose as previously discussed. Four layered CNN model is employed but size of filter is (8, 7, 6, 3), number of epochs are 25 and error rate is dropping at each epoch (See Figure:4.2(a)), 0.0002% and 0.18% are training and validation error rates respectively, accuracy (See Figure:4.2(b)) for training is 1% and for validation set is 0.97%.



[a] Loss for Degree of Characters One

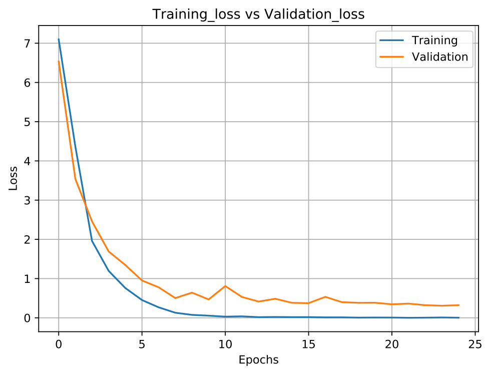


[b] Accuracy for Degree of Characters One

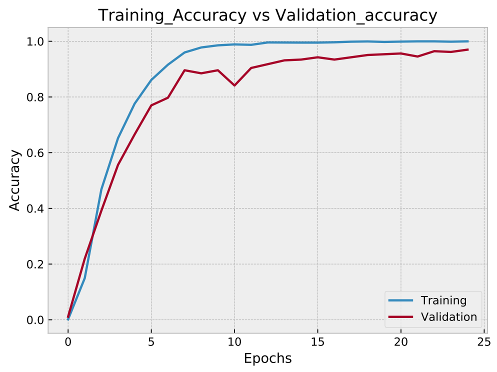
Figure 4.2: Degree-1 at Level-1

4.2.2 Degree-2

At Degree of character two dataset is increased with 4, 590 images (See Table:4.1) of 100*100, for training set error rate is 0.005% and for validation it is 0.3% (See Figure:4.3(a)), by adding a dropout up-to 0.25 variance in dataset is decreased, achieved validation accuracy (See Figure:4.3(b)) of model is 0.98%.



[a] Loss for Degree of Characters Two

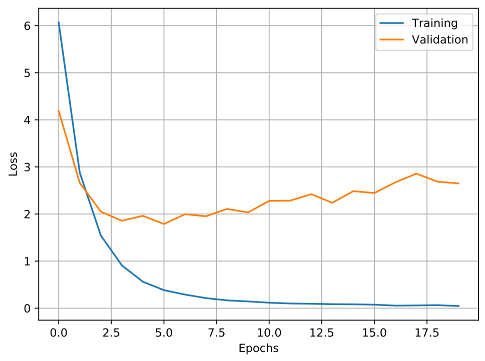


[b] Accuracy for Degree of Characters Two

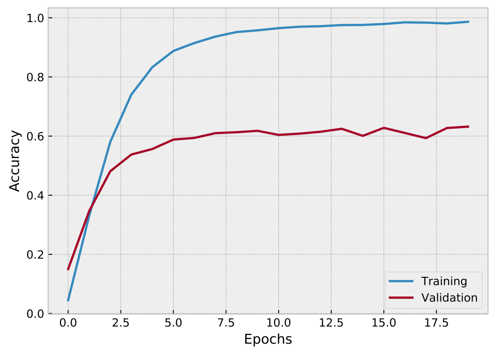
Figure 4.3: Degree-2 at Level-1

4.2.3 Degree-3

At Degree of character three dataset is increased with 18, 900 images (See Table:4.1) of 100*100, with number of classes 1, 260, resnet is implemented (See Table. 4.8), for training set error rate is 0.003% and for validation it is 0.9% (See Figure:4.4(a)), achieved validation accuracy (See Figure:4.4(b)) of model is 0.63%.



[a] Loss for Degree of Characters Three



[b] Accuracy for Degree of Characters Three

Figure 4.4: Degree-3 at Level-1

4.3 Summary of Results

Results of proposed method are explained in this section for each generated dataset. First table discusses dataset specifications for level, at level-0 dataset is trained from 22,000 to 66,000 images and produced promising results. Every dataset is further trained with different parameters but results are reported for specific parameters.

4.3.1 Dataset Specification

Generated dataset specifications are mentioned in below table along the final classes of every dataset which we used for testing and training purpose number of fonts were different at two levels for better training degree-1 is also trained with almost 93 more fonts and at level-0 our images exceeded to almost 66, 000.

Table 4.1: Dataset Specification

Dataset	No. of Classes	No. of Fonts	No. of Images
Level-0			
Level-0	3	15	19, 435
Level-1			
Degree-1	19	15	285
Degree-2	150	15	2, 250
Degree-3	1260	15	18, 900

4.3.2 CNN Architecture Configuration

CNN keras model is experimented with almost all available optimizers such as Adam, SGD, Adagrad, Adamax and Nadam also activation functions include, sigmoid, tanh, relu and advanced activation function, LeakyRelu, PReLU, ELU, SReLU, pooling layers Averagepooling, Maxpooling, Globalmaxpooling, Globalaveragepooling, but our model produced some best results with optimizer RMSprop and activation function SReLU with MaxPooling layer only. Learning rate for every model was fixed. In below table (See 4.3)

Table 4.2: CNN Architecture Configuration Part-1

Dataset	Epoch	Activation Layer	Learning rate	Optimizer
Level-0	5	SReLU	lr=le-3	RMSProp
Level-1				
Degree-1	25	SReLU	lr=le-3	RMSProp
Degree-2	25	SReLU	lr=le-3	RMSProp
Degree-3	25	SReLU	lr=le-3	RMSProp

class-weights are assigned to balance the classes as at level-0 number of samples in three classes are imbalance.

Table 4.3: CNN Architecture Configuration Part-2

Dataset	No. of Filters	Regularizer	Loss	Class Weights
Level-0	48	0.2	Categorical_crossentropy	1:350, 2:30, 3:10
Level-1				
Degree-1	128	0.2	Categorical_crossentropy	Nill
Degree-2	128	0.2	Categorical_crossentropy	Nill
Degree-3	128	0.2	Categorical_crossentropy	Nill

4.3.3 Recognition Rate

Reported recognition rates includes F-1 score, precision and recall due to imbalance dataset at level-0 class weights are assigned, two layered model of level-0 achieved 0.99% accuracy for both training and test with F-1 score of 0.99% as well. For degree-1 and 2 achieved accuracy is 0.97 and 0.98% respectively as number of classes are increased at degree-3 so achieved accuracy is reduced to 0.68%.

Table 4.4: Recognition Rate

Dataset	**Accuracy**	**Precision**	**Recall**	**F-1 Score**
Level-0	0.99%	0.99%	0.99%	0.99%
		Level-1		
Degree-1	0.968%	0.98%	0.97%	0.97%
Degree-2	0.98 %	0.94 %	0.93 %	0.93%
Degree-3	0.63 %	0.72%	0.63%	0.63%

4.3.4 Model Summary

CNN keras two layer model summary of level-0 (See Table.4.5) is in detail. Two convolutional layers with max_pool layer and activation layer are added, dense layer is used as fully connected layer for each class same is for degree-1 (See Table.4.6). (See Table 4.8) is creating residual connection of level-1 for degree of characters three. The number of parameters are calculated through following formula.

For first Conv2D layer number of parameters:

$$(filterhight * filterwidth * inputimagechannel + 1) * numberoffilters \tag{4.1}$$

Similarly for second Conv2D layer number of parameters:

$$(filterhight * filterwidth * nooffilterspreviouslayer + 1) * numberoffilters \tag{4.2}$$

At level-0 convolutional neural netwrok model summary explains the number of hidden layers along the other layers and total calculated parameters. The result we achived at this level-0 is 0.99 % with by implementing two CNN layers.

Table 4.5: CNN Model Summary Level-0

Layers	**Output Shape**	**No of Param**
Input Layer	(None, 100, 100, 1)	0
Conv2D	(None, 100, 100, 16)	160
SReLU	(None, 100, 100, 16)	640000
MaxPooling	(None, 50, 50, 16)	0
Conv2D	(None, 50, 50, 32)	4640
SReLU	(None, 50, 50, 32)	320000
MaxPooling	(None, 25, 25, 32)	0
Dropout	(None, 25, 25, 32)	0
Dense	(None, 4)	80004
Activation	(None, 4)	0

Table (4.6) is model summary at level-1 for degree-1 and Table (4.7) for degree-2. Same model is implemented for every dataset. but the number of classes were increasing with each degree based on their components.

Table 4.6: CNN Model Summary Level-1 for Degree-1

Layers	**Output Shape**	**No of Param**
Input Layer	(None, 100, 100, 1)	0
Conv2D	(None, 100, 100, 16)	1312
SReLU	(None, 100, 100, 16)	640000
MaxPooling	(None, 50, 50, 16)	0
Conv2D	(None, 50, 50, 32)	32800
SReLU	(None, 50, 50, 32)	320000
MaxPooling	(None, 25, 25, 32)	0
Conv2D	(None, 25, 25, 64)	100416
SReLU	(None, 25, 25, 64)	160000
MaxPooling	(None, 12, 12, 64)	0
Conv2D	(None, 12, 12, 128)	73856
SReLU	(None, 12, 12, 128)	73728
MaxPooling	(None, 6, 6, 128)	0
Dropout	(None, 6, 6, 128)	0
Flatten	(None, 4608)	0
Dense	(None, 37)	170533

Table (4.8) this model is based on residual convolutional neural network, due to increased number of classes in this dataset result was diminishing, and as previously discussed

Table 4.7: CNN Model Summary Level-1 for Degree-1

Layers	**Output Shape**	**No of Param**
Input Layer	(None, 100, 100, 1)	0
Conv2D	(None, 100, 100, 16)	1312
SReLU	(None, 100, 100, 16)	640000
MaxPooling	(None, 50, 50, 16)	0
Conv2D	(None, 50, 50, 32)	32800
SReLU	(None, 50, 50, 32)	320000
MaxPooling	(None, 25, 25, 32)	0
Conv2D	(None, 25, 25, 64)	100416
SReLU	(None, 25, 25, 64)	160000
MaxPooling	(None, 12, 12, 64)	0
Conv2D	(None, 12, 12, 128)	73856
SReLU	(None, 12, 12, 128)	73728
MaxPooling	(None, 6, 6, 128)	0
Dropout	(None, 6, 6, 128)	0
Flatten	(None, 4608)	0
Dense	(None, 150)	170533
Activation	(None, 150)	0

increased number of layers causes degrading of accuracy. Residual nets deals with vanishing gradient problem and it produced good accuracy for degree-3.

Table 4.8: Residual CNN Model Summary at Level-1 for Degree-3

Layers	**Output Shape**	**Param**	**Connected to**
Input Layer	(None, 100, 100, 1)	0	
Conv2D	(None, 100, 100, 1)	10	Input[0][0]
Add	(None, 100, 100, 1)	0	Input[0][0]-Conv2D[0][0]
Conv2D	(None, 100, 100, 8)	808	Add[0][0]
MaxPooling2D	(None, 50, 50, 8)	0	Conv2D[0][0]
SReLU	(None, 50, 50, 8)	80000	MaxPooling2D[0][0]
Conv2D	(None, 50, 50, 16)	10384	SReLU[0][0]
MaxPooling2D	(None, 25, 25, 16)	0	Conv2D[0][0]
SReLU	(None, 25, 25, 16)	40000	MaxPooling2D[0][0]
Flatten	(None, 1152)	0	SReLU[0][0]
Dense	(None, 1260)	1310961	Flatten[0][0]
Activation	(None,1260)	0	Dense[0][0]

Chapter 5

Conclusions and Future Work

Urdu is national language of Pakistan and recently declared as official language. It is a popular and widely spoken language of Pakistan, India and other South Asian countries including some western countries as well. Urdu follows two different writing styles Naskh and Nastaliq while Nastaliq is different in terms of diagonality. Urdu is rather complex as its morphology and syntax structure foundations are mixture of different languages. Character classification can be implemented only if segmentation technique is efficient, but due to context sensitivity in Urdu, segmentation-based recognition often results with high error rate. However, character classification has implemented for the many western languages, but Urdu research community have shifted their focus to the ligature recognition as it takes minimal segmentation.

In this research, component-based recognition for Urdu Naskh text is implemented through convolutional neural networks a deep learning-based architecture. Urdu character recognition has been a complex task to handle as per various limitations of a language, segmentation free recognition so far produced good results as compared to segmentation-based strategies.

Our proposed approach is component-based method trained on Urdu text which identifies each ligature as a separate component and recognizes it whether component is combination of one character, two or three. Component separation is done by using two pass connected component algorithm to reduce the number of classes, which label pixels based on their neighbor pixels, if pixels belong to the same region same label is assigned, after separating un-necessary components, convolutional neural network is used for learning and classification. Learning and classification is further divided into two levels where at first level we classify the number of characters in a component and on the rest of the levels individual components are being recognized. Convolutional neural network is deep learning architecture which have property of automatic feature learning, instead of defining features manually it automatically extracts features based on weights. Dataset of Urdu ligature is a text-based ligature generated through permutation for three characters and trained through statistical technique of deep learning we have also used residual convolutional neural networks for better learning at level-1 where dataset was small.

Implementation of component recognition is employed by two layered CNN architecture which is used for feature extraction of 18, 900 Urdu text ligature images of 100 * 100 as recognition of component is divided into two levels and at level-0 our goal is to classify component based on character it comprises. Error rate is decreasing with increasing number of epochs, number of epochs are 20, training error rate is 0.0013% while validation error rate is 0.0003%. Accuracy achieved for training dataset is 0.9998% and for validation its 0.99% as dataset is imbalance in this scenario for which class weights are used to assign weights to the classes and besides accuracy, precision, recall and F-1 score is reported as well which is 0.99% in this case. This is the perfect loss of trained convolutional neural network due to large dataset.

As we have implemented hierarchical neural network for component recognition where after level-0, at level-1 component recognition takes place. At level-0 obtained results are 0.99 % for recognition of category of component whether component belongs to degree-1, degree-2 or degree-3. Four layered CNN model is employed at level-1 for degree of characters one, number of epochs are 25 and error rate is dropping at each epoch, 0.0002% and 0.18% are training and validation error rates respectively, accuracy for training is 1% and for validation set is 0.97%. At degree two for training set error rate is 0.005% and for validation it is 0.3%, by adding a dropout up-to 0.25 variance in dataset is decreased, achieved validation accuracy of model is 0.98%. At degree three for training set error rate is 0.003% and for validation it is 0.9%, achieved validation accuracy of model is

0.63%. At degree three resnet is implemented, for training set error rate is 0.003% and for validation it is 0.9%, achieved validation accuracy of model is 0.63%.

CNN keras two layer model summary of level-0 is implemented. Two convolutional layers with max_pool layer and activation layer are added, dense layer is used as fully connected layer for each class same is for degree-1 and degree-2. Further, creating residual connection due increased number of classes of level-1 for degree of characters three.

This research is part of a huge project (See Figure.(5.1)) where targeted documents are specifically hand written Naskh font of Urdu language. Most of the existing Urdu documents are in Nastaleeq font. Urdu language research is moved from character to ligature. Our approach reduces the ligature into component after this part of a process, project where a component is being recognized future work moves to the recognition of whole word by implementing a model and generating a lexicon. Combining two recognized components, based on the context of a sentence to generate a word for which a lexicon will be implemented initially.

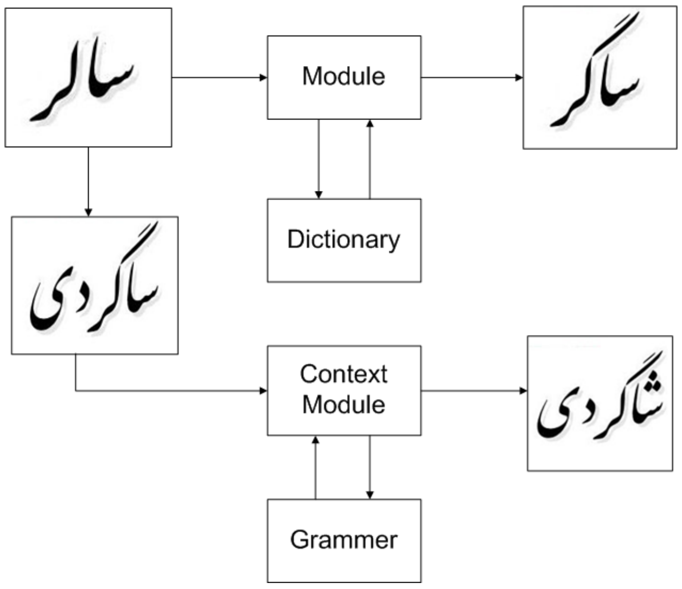


Figure 5.1: Model of Proposed Future Work

References

[1] I. Ahmad, X. Wang, R. Li, M. Ahmed, and R. Ullah, “Line and ligature segmentation of urdu nastaleeq text,” *IEEE Access*, 2017.

[2] N. H. Khan, A. Adnan, and S. Basar, “Urdu ligature recognition using multi-level agglomerative hierarchical clustering,” *Cluster Computing*, pp. 1–12, 2017.

[3] A. R. Khan and Z. Mohammad, “A simple segmentation approach for unconstrained cursive handwritten words in conjunction with the neural network,” *International Journal of Image Processing*, vol. 2, no. 3, pp. 29–35, 2008.

[4] M. Kavianifar and A. Amin, “Preprocessing and structural feature extraction for a multi-fonts arabic/persian ocr,” in *Document Analysis and Recognition, 1999. ICDAR’99. Proceedings of the Fifth International Conference on.* IEEE, 1999, pp. 213–216.

[5] M. Amrouch and M. Rabi, “Deep neural networks features for arabic handwriting recognition,” in *International Conference on Advanced Information Technology, Services and Systems.* Springer, 2017, pp. 138–149.

[6] T. Ali, T. Ahmad, and M. Imran, “Uocr: A ligature based approach for an urdu ocr system,” in *Computing for Sustainable Global Development (INDIACom), 2016 3rd International Conference on.* IEEE, 2016, pp. 388–394.

[7] S. Naz, A. I. Umar, R. Ahmad, I. Siddiqi, S. B. Ahmed, M. I. Razzak, and F. Shafait, “Urdu nastaliq recognition using convolutional–recursive deep learning,” *Neurocomputing*, vol. 243, pp. 80–87, 2017.

[8] I. Ahmad, X. Wang, Y. hao Mao, G. Liu, H. Ahmad, and R. Ullah, "Ligature based urdu nastaleeq sentence recognition using gated bidirectional long short term memory," *Cluster Computing*, pp. 1–12, 2017.

[9] N. Sabbour and F. Shafait, "A segmentation-free approach to arabic and urdu ocr." in *DRR*, 2013, p. 86580N.

[10] "https://sebastianraschka.com/articles/2015_singlelayer_neurons.html."

[11] "http://cs231n.github.io/neural-networks-1/."

[12] "https://mashimo.wordpress.com/tag/back-propagation/."

[13] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *The Journal of Machine Learning Research*, vol. 15, no. 1, pp. 1929–1958, 2014.

[14] "https://www.rsipvision.com/exploring-deep-learning/."

[15] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[16] S. Shabbir and I. Siddiqi, "Optical character recognition system for urdu words in nastaliq font," *International Journal of Advanced Computer Science and Applications (IJACSA)*, vol. 7, no. 5, pp. 567–576, 2016.

[17] K. Riaz, "Baseline for urdu ir evaluation," in *Proceedings of the 2nd ACM workshop on Improving non english web searching*. ACM, 2008, pp. 97–100.

[18] S. B. Ahmed, S. Naz, S. Swati, and M. I. Razzak, "Handwritten urdu character recognition using 1-dimensional blstm classifier," *arXiv preprint arXiv:1705.05455*, 2017.

[19] F. Adeeba and S. Hussain, "Experiences in building urdu wordnet," in *Proceedings of the 9th workshop on Asian language resources*, 2011, pp. 31–35.

[20] D. A. Satti and K. Saleem, “Complexities and implementation challenges in offline urdu nastaliq ocr,” in *Proceedings of the Conference on Language and Technology*, 2012, pp. 85–91.

[21] N. Durrani and S. Hussain, “Urdu word segmentation,” in *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*. Association for Computational Linguistics, 2010, pp. 528–536.

[22] R. G. Casey and E. Lecolinet, “A survey of methods and strategies in character segmentation,” *IEEE transactions on pattern analysis and machine intelligence*, vol. 18, no. 7, pp. 690–706, 1996.

[23] W.-S. W. Lian and D. P. Pozefsky, “Heuristic-based ocr post-correction for smart phone applications,” *University of North Carolina*, 2009.

[24] M. Asad, A. S. Butt, S. Chaudhry, and S. Hussain, “Rule-based expert system for urdu nastaleeq justification,” in *Multitopic Conference, 2004. Proceedings of INMIC 2004. 8th International*. IEEE, 2004, pp. 591–596.

[25] M. Blumenstein and B. Verma, “Neural-based solutions for the segmentation and recognition of difficult handwritten words from a benchmark database,” in *Document Analysis and Recognition, 1999. ICDAR’99. Proceedings of the Fifth International Conference on*. IEEE, 1999, pp. 281–284.

[26] D. Cai and H. Zhao, “Neural word segmentation learning for chinese,” *arXiv preprint arXiv:1606.04300*, 2016.

[27] M. Zhang, Y. Zhang, and G. Fu, “Transition-based neural word segmentation.” 2016.

[28] Y. Niu and L. Li, “An improved chinese segmentation algorithm based on segmentation dictionary,” in *Computer Technology and Development, 2009. ICCTD’09. International Conference on*, vol. 1. IEEE, 2009, pp. 184–187.

[29] M. Akram and S. Hussain, “Word segmentation for urdu ocr system,” in *Proceedings of the 8th Workshop on Asian Language Resources, Beijing, China*, 2010, pp. 88–94.

[30] G. S. Lehal, “Ligature segmentation for urdu ocr,” in *Document Analysis and Recognition (ICDAR), 2013 12th International Conference on*. IEEE, 2013, pp. 1130–1134.

[31] M. M. Altuwaijri and M. A. Bayoumi, “Arabic text recognition using neural networks,” in *Circuits and Systems, 1994. ISCAS’94., 1994 IEEE International Symposium on*, vol. 6. IEEE, 1994, pp. 415–418.

[32] S. S. Ahranjany, F. Razzazi, and M. H. Ghassemian, “A very high accuracy handwritten character recognition system for farsi/arabic digits using convolutional neural networks,” in *Bio-Inspired Computing: Theories and Applications (BIC-TA), 2010 IEEE Fifth International Conference on*. IEEE, 2010, pp. 1585–1592.

[33] M. Jain, M. Mathew, and C. Jawahar, “Unconstrained scene text and video text recognition for arabic script,” *arXiv preprint arXiv:1711.02396*, 2017.

[34] R. Ahmad, S. Naz, M. Z. Afzal, S. H. Amin, and T. Breuel, “Robust optical recognition of cursive pashto script using scale, rotation and location invariant approach,” *PloS one*, vol. 10, no. 9, p. e0133648, 2015.

[35] U. Pal and A. Sarkar, “Recognition of printed urdu script,” in *Document Analysis and Recognition, 2003. Proceedings. Seventh International Conference on*. IEEE, 2003, pp. 1183–1187.

[36] I. Shamsher, Z. Ahmad, J. K. Orakzai, and A. Adnan, “Ocr for printed urdu script using feed forward neural network,” in *Proceedings of World Academy of Science, Engineering and Technology*, vol. 23, 2007, pp. 172–175.

[37] N. Shahzad, B. Paulson, and T. Hammond, “Urdu qaeda: recognition system for isolated urdu characters,” in *Proceedings of the IUI Workshop on Sketch Recognition, Sanibel Island, Florida*, 2009.

[38] Z. Ahmad, J. K. Orakzai, and I. Shamsher, “Urdu compound character recognition using feed forward neural networks,” in *Computer Science and Information Technology, 2009. ICCSIT 2009. 2nd IEEE International Conference on*. IEEE, 2009, pp. 457–462.

[39] M. W. Sagheer, C. L. He, N. Nobile, and C. Y. Suen, “Holistic urdu handwritten word recognition using support vector machine,” in *Proceedings of the 2010 20th International Conference on Pattern Recognition.* IEEE Computer Society, 2010, pp. 1900–1903.

[40] A. Abidi, I. Siddiqi, and K. Khurshid, “Towards searchable digital urdu libraries-a word spotting based retrieval approach,” in *Document Analysis and Recognition (ICDAR), 2011 International Conference on.* IEEE, 2011, pp. 1344–1348.

[41] K. Khan, R. Ullah, N. A. Khan, and K. Naveed, “Urdu character recognition using principal component analysis,” *International Journal of Computer Applications*, vol. 60, no. 11, 2012.

[42] S. Raj, Z. Rehman, S. Rauf, R. Siddique, and W. Anwar, “An artificial neural network approach for sentence boundary disambiguation in urdu language text.” *International Arab Journal of Information Technology (IAJIT)*, vol. 12, no. 4, 2015.

[43] Z. Ahmad, J. K. Orakzai, I. Shamsher, and A. Adnan, “Urdu nastaleeq optical character recognition,” in *Proceedings of world academy of science, engineering and technology*, vol. 26. Citeseer, 2007, pp. 249–252.

[44] T. Nawaz, S. Naqvi, H. ur Rehman, and A. Faiz, “Optical character recognition system for urdu (naskh font) using pattern matching technique,” *International Journal of Image Processing (IJIP)*, vol. 3, no. 3, p. 92, 2009.

[45] D. A. Satti, “Offline urdu nastaliq ocr for printed text using analytical approach,” Ph.D. dissertation, MS Thesis report, Quaid-i-Azam University, Islamabad, 2013.

[46] S. T. Javed, S. Hussain, A. Maqbool, S. Asloob, S. Jamil, and H. Moin, “Segmentation free nastalique urdu ocr,” *World Academy of Science, Engineering and Technology*, vol. 46, pp. 456–461, 2010.

[47] S. Naz, A. I. Umar, R. Ahmad, S. B. Ahmed, S. H. Shirazi, and M. I. Razzak, “Urdu nastaliq text recognition system based on multi-dimensional recurrent neural network and statistical features,” *Neural Computing and Applications*, vol. 28, no. 2, pp. 219–231, 2017.

[48] I. Ahmad, X. Wang, R. Li, and S. Rasheed, “Offline urdu nastaleeq optical character recognition based on stacked denoising autoencoder,” *China Communications*, vol. 14, no. 1, pp. 146–157, 2017.

[49] B. Epshtein, E. Ofek, and Y. Wexler, “Detecting text in natural scenes with stroke width transform,” in *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*. IEEE, 2010, pp. 2963–2970.

[50] G. Werner, “Text detection in natural scenes with stroke width transform,” *Ben Gurion University, Israel*, 2013.

[51] K. Wu, E. Otoo, and K. Suzuki, “Optimizing two-pass connected-component labeling algorithms,” vol. 12, no. 2. Springer, 2009, pp. 117–135.

[52] K. Gurney, *An introduction to neural networks*. CRC press, 2014.

[53] W. S. McCulloch and W. Pitts, “A logical calculus of the ideas immanent in nervous activity,” *The bulletin of mathematical biophysics*, vol. 5, no. 4, pp. 115–133, 1943.

[54] Y. LeCun, Y. Bengio, and G. Hinton, “Deep learning,” *Nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[55] Q. Akram, S. Hussain, F. Adeeba, S. Rehman, and M. Saeed, “Framework of urdu nastalique optical character recognition system,” in *the Proceedings of Conference on Language and Technology.(CLT 14), Karachi, Pakistan*, 2014.

[56] S. Yousfi, S.-A. Berrani, and C. Garcia, “Alif: A dataset for arabic embedded text recognition in tv broadcast,” in *Document Analysis and Recognition (ICDAR), 2015 13th International Conference on*. IEEE, 2015, pp. 1221–1225.

[57] M. Pechwitz, S. S. Maddouri, V. Märgner, N. Ellouze, H. Amiri *et al.*, “Ifn/enit-database of handwritten arabic words,” in *Proc. of CIFED*, vol. 2. Citeseer, 2002, pp. 127–136.

[58] K. Lagally, “Arabtex—typesetting arabic with vowels and ligatures,” *EuroTEX*, vol. 92, pp. 153–172, 1992.

[59] F. Slimane, R. Ingold, S. Kanoun, A. M. Alimi, and J. Hennebert, "Database and evaluation protocols for arabic printed text recognition," *DIUF-University of Fribourg-Switzerland*, 2009.

[60] A. Qurrat-ul, A. Niazi, F. Adeeba, S. Urooj, S. Hussain, and S. Shams, "A comprehensive image dataset of urdu nastalique document images," in *Proceedings of the conference on language and technology*, 2016, pp. 81–88.

[61] N. Kalchbrenner, E. Grefenstette, and P. Blunsom, "A convolutional neural network for modelling sentences," *arXiv preprint arXiv:1404.2188*, 2014.

[62] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in neural information processing systems*, 2012, pp. 1097–1105.

[63] Y. Kim, "Convolutional neural networks for sentence classification," *arXiv preprint arXiv:1408.5882*, 2014.