DEVELOPMENT OF FRAMEWORK FOR CLASSIFICATION AND ARCHIVING HISTORICAL MANUSCRIPT IMAGES

A Thesis Submitted to

Delhi Technological University

for the Award of Degree of

Doctor of Philosophy

In

Electronics and Communication Engineering

By

ENOCK OSORO OMAYIO

(Enrollment No.: 2K17/PHDEC/13)

Under the Supervision of

Dr. INDU SREEDEVI

Professor and Dean (Student Welfare)

Dr. JEEBANANDA PANDA

Professor in Department of Electronics and Communication Engineering

Department of Electronics and Communication Engineering

Delhi Technological University (Formerly DCE)

Bawana Road, Delhi - 110042, India

January 2023

DEVELOPMENT OF FRAMEWORK FOR CLASSIFICATION AND ARCHIVING HISTORICAL MANUSCRIPT IMAGES

A Thesis Submitted to

Delhi Technological University

for the Award of Degree of

Doctor of Philosophy

In

Electronics and Communication Engineering

By

ENOCK OSORO OMAYIO

(Enrollment No.: 2K17/PHDEC/13)

Under the Supervision of

Dr. INDU SREEDEVI

Professor and Dean (Student Welfare)

Dr. JEEBANANDA PANDA

Professor in Department of Electronics and Communication Engineering

Department of Electronics and Communication Engineering

Delhi Technological University (Formerly DCE)

Bawana Road, Delhi - 110042, India

January 2023

DECLARATION

I hereby declare that the work which is being presented in the thesis entitled, **"Development of Framework for Classification and Archiving Historical Manuscript Images"** in partial fulfillment of the requirements for the award of the degree of Doctor of Philosophy and submitted in the Department of Electronics and Communication Engineering of Delhi Technological University, Delhi is an authentic record of my own work carried out during the period from 2017 to 2022 under the supervision of Prof Sreedevi Indu, and Prof Jeebananda Panda, both Professors in the Department of Electronics and Communication Engineering, Delhi Technological University, Delhi, India. The content of this thesis has not been submitted either in part or whole to any other university or institute for the award of any degree or diploma.

Enock Osoro Omayio

CERTIFICATE

This is to certify that the thesis titled, "**Development of Framework for Classification and Archiving Historical Manuscript Images**" being submitted by Enock Osoro Omayio to the Department of Electronics and Communication Engineering, Delhi Technological University, Delhi, for the award of the degree of Doctor of Philosophy, is a record of bonafide research work carried out by him under our guidance and supervision. In our opinion, the thesis has reached the standards fulfilling the requirements of the regulations relating to the degree. The results contained in this thesis have not been submitted to any other university or institute for the award of any degree or diploma.

Prof. Indu Sreedevi
Professor and Dean (Student Welfare)
Department of Electronics and Communication Engineering
Delhi Technological University
Delhi 110042

Prof. Jeebananda Panda
Professor
Department of Electronics and Communication Engineering
Delhi Technological University
Delhi 110042

DEDICATION

I graciously dedicate this thesis to the Almighty God, my dear loving wife Beatrice Gesare, our dear sons Gamaliel Ondieki and Gabriel Omayio, and daughter Gracelyn Monyenye.

ACKNOWLEDGMENTS

If i have seen further, it is by standing on the shoulders of the Giants.

—— *Sir Isaac Newton*

I gracefully thank the Almighty God for His providence and enablement that has seen me complete this PhD course. The PhD journey has been challenging and full of learning opportunities and experiences that has helped me up my research acumen. It is by God's grace I have come thus far.

In a special way, I salute my research guides Prof Sreedevi Indu and Prof Jeebananda Panda, both distinguished professors in Electronics and Communication Engineering Department at Delhi Technological University in India. They mentored me scholarly from the time I was admitted to PhD scholarship. Their advice on conducting research, writing research paper and articles is applauded in great measure. They walked me down the research path with tremendous interest, concern, effort, and enthusiasm. I have seen and reached further because I stand on the shoulders of two giants, Prof Indu Sreedevi and Prof Jeebananda Panda. To them I say, may God bless you lavishly with good health, and long life full of God's favours. In addition, i deeply appreciate the head of the department of Electronics and Communication Engineering (ECE), the faculty members and non-teaching staff of ECE department at Delhi Technological University in India for their immense support and encouragement throughout the research period.

To my lovely and loving wife Beatrice Gesare, I appreciate deeply for her love, her unceasing prayers, and patience for the long times I was away in college immersed in research and pursuing the studies. Her constant encouragement and support kept me going during tough phases of research. She was my strength and driving power in this auspicious endeav-

our. To our loving sons Gamaliel and Gabriel, and daughter Gracelyn, I always found joy, happiness and relaxing moments. Their sonly and daughterly love, laughs, and enthusiasm always livened my spirits during research periods. They patiently endured my prolonged absence from home as I was away studying. This always reminded me to remain focused till completion. May God richly bless their lives lively going forward.

My loving parents Robert Omayio and Agnes Kwamboka, and parents-in-law Peter Moseti and Priscah Basweti were all beacons of hope and inspiration to me throughout my study journey. Their prayers and encouragements kept me strong and steady throughout. They constantly checked on me and urging me never to give up till completion. I am indebted to say a big thank you for your emotional, moral and inspirational support.

To my siblings George, Zipporah, Sarah, Susan, Stephen, Edward, and Jackline Bikoro, I also convey a great appreciation for your prayers. God bless you abundantly for the moral support you gave me. To my in-laws Hellen, Joyce, Robert, Rachel, Elkanah, and Sarah, I heartly acknowledge your prayers and good wishes with a thankful heart. You were a constant support to me throughout the study period.

Finally, I take cognizance of my PhD colleagues like Ajay Kaushik, Shashank, Ishu Tomar, Rajiv Yadav, and Gebrekiros Gebreyesus for their constructive collaboration and support in our respective research areas. It was good times to have travelled together in this PhD journey.

The PhD journey was a good undertaking worth the time, efforts, and will.
To God be the glory
Amen.

Enock Osoro Omayio
New Delhi
January 2023

ABSTRACT

Historical manuscripts are valuable resources for historical information about the distant past. From them culture, education, and ways of life in the past can be gleaned. Due to advancement of Information Communication and Technology (ICT), most of the historical manuscripts have been digitized via scanning devices to electronic formats like digital images. This has resulted in large amounts of historical manuscripts available to public as digital images and other electronic forms. Historical manuscript images (HMI) are easier to manage (by sharing, handling, storage, and processing) compared to actual manuscript documents. In addition, this helps to preserve actual historical manuscripts since they are seldom needed physically.

Due to the proliferation of large amounts of HMI, their management and processing is the main focus. HMI management involves a range of tasks and processes carried out on HMI like curation, provenance, indexing and archiving, storage, restoration, retrieval, and classification among others. This thesis focuses on development of computer-based framework to index, archive, and classify HMI. First of all, a number of pre-processing tasks are carried out to enhance visual quality of HMI and in turn increase output performance. The pre-processing tasks carried out include denoising, binarization, word segmentation and word image normalization.

Due to degradations in most of HMI, a model-based binarization technique is proposed to enhance them. In this technique, HMI pixels are modelled to foreground and background pixels by training multilayer perceptron (MLP) classifier using various handcrafted features with high discriminating powers. The features are extracted from HMI.

A component tracing and association (CTA) technique has been developed for efficient word

segmentation of HMI. The merit of the method is in segmenting overlapping and crossing words. Using the concept that short sections of a continuous stroke joined at a common point are symmetric or near symmetric about the common joining point, crossing strokes are identified and separated using ($MD-DTW_D$) multi-dimensional dynamic time warping with dependence. method.

A segmentation-based handwritten word spotting (HWS) technique has been developed for indexing HMI. Integral histogram of oriented displacement (IHOD) feature descriptor is used to develop MLP-based HWS system. IHOD descriptor is obtained by computing displacements of foreground pixels w.r.t centers of their respective $m \times m$ cells where $m = 15$ pixels. Cells are obtained by sub-dividing entire HMI.

A fragmented long short-term memory (Frag-LSTM) method is proposed for language identification (LID) of textual content of HMI. 3 LSTM networks are used to learn and extract local and global features from input text word. A combined feature vector is obtained by concatenating global and local features. This combined feature vector is then used for LID.

Bi-directional fragment network (BiD-FragNet) is proposed for prediction of era or production time of HMI. BiD-FragNet consists of 2 convolution neural network (CNN)-based channels; main and fragment channels. The main channel learns and extracts global features by processing full patches of HMI. Fragment channel is used to learn and extract local features by processing fragments (sub-patches) of HMI. Both channels share information in both directions at various levels. Global and local features learnt are then concatenated to one feature vector which is used with classification layer to give final classification output. Classification output is obtained by voting and averaging schemes.

Funnelling ensemble method for writer identification (FEM-WI) has been proposed for HMI. It is a 2-level system of classifier ensembles. in this system, first, 5 newly proposed features (also called base features) are extracted from segmented handwritten words. In level 1, each feature is used to train individual base classifier (MLP). Meta features are then obtained as outputs of level 1 (base) classifiers via k-fold cross validation (KFCV) method. A single level 2 meta classifier that gives final output is trained using the meta features. FEM-WI works by leveraging on different base features for same input word image funnelled to a common

feature space in level 2 classifier. Thus, writer identification of query word using any one of the base features benefits from all other features used to train meta classifier, hence giving improved output performance.

Table of Contents

Page

DECLARATION **i**

CERTIFICATE **ii**

DEDICATION **iii**

ACKNOWLEDGMENTS **iv**

ABSTRACT **vi**

LIST OF FIGURES **xv**

LIST OF TABLES **xx**

1 INTRODUCTION **1**

1.1 Need for historical manuscript studies and associated challenges 1

1.2 Computer-based management of historical manuscripts and its issues 2

1.3 Problem statement 3

1.4 Scope and objectives of the thesis 5

1.5 Contribution and Thesis Layout 6

2 LITERATURE REVIEW **9**

2.1 Introduction 9
2.2 Restoration methods for degraded historical manuscript images 9
2.2.1 Threshold-based binarization methods 10
2.2.2 Non-threshold-based binarization methods 11
2.2.3 Hybrid binarization methods for historical manuscript images 11
2.2.4 Handcrafted features-based binarization methods 11
2.2.5 Deep learning-based binarization methods 12
2.3 Word spotting techniques for historical methods 12
2.4 Auto-Writer identification methods for historical manuscript images 13
2.4.1 Handcrafted features-based writer identification methods 14
2.4.2 Deep learning-based (DLB) writer identification methods 14
2.5 Historical manuscript dating (HMD) techniques 15
2.5.1 Physical techniques for HMD 15
2.5.2 Paleographical methods for HMD 16
2.5.3 Computer-based techniques of HMD 17
2.6 Language identification methods for historical manuscripts 18
2.6.1 Word-based LID methods 19
2.6.2 N-gram-based LID methods 19
2.6.3 Learning-based LID methods 20

3 MANUSCRIPT ENHANCEMENT BY BINARIZATION METHOD 21

3.1 Introduction 21
3.2 Binarization of HMI 22
3.3 Problem statement 22
3.4 MLP-based binarization method 23
3.4.1 Pre-processing 24

3.4.2 Feature extraction 24
3.4.3 Model development 27
3.5 Evaluation of MLP-based binarization method 28
3.6 Conclusion 36

4 HANDWRITTEN WORD SPOTTING FOR INDEXING HISTORICAL MANUSCRIPTS 37

4.1 Introduction 37
4.2 Word spotting 37
4.3 Problem statement 40
4.4 Word segmentation 40
4.4.1 Line segmentation 42
4.4.2 Core word segmentation 45
4.4.3 Multi-dimensional dynamic time warping with dependence ($MD - DTW_D$) 48
4.4.4 Junction branch association (JBA) method 50
4.4.5 Full word segmentation 52
4.5 Handwritten word spotting (HWS) 54
4.5.1 Pre-processing 55
4.5.2 Word image segmentation 55
4.5.3 Normalization 56
4.5.4 IHOD feature extraction 57
4.5.4.1 Word image standardization 58
4.5.4.2 IHOD computation 58
4.5.4.3 IHOD vector size 62
4.5.5 Training 63

4.6 Experimental results and discussion 65
4.6.1 Evaluation of CTA technique for word segmentation 65
4.6.2 Evaluation of handwritten Word spotting (HWS) model 71
4.6.2.1 Evaluation Datasets 71
4.6.2.2 Data augmentation 71
4.6.2.3 Evaluation metrics 72
4.6.2.4 Evaluation of handwritten Word spotting (HWS) model . . 73
4.7 Conclusion 76

5 LANGUAGE IDENTIFICATION FOR HISTORICAL MANUSCRIPT DOCUMENTS 78

5.1 Introduction 78
5.2 Language Identification (LID) 79
5.3 Problem statement 80
5.4 Fragmented LSTM for Language Identification (Frag-LSTM-LID) 80
5.4.1 Fragmentation 80
5.4.2 Pre-processing 81
5.4.3 Working mechanism of LSTM 83
5.4.4 Training of LSTM 86
5.5 Experimental Results 87
5.5.1 Evaluation Datasets 87
5.5.2 Evaluation metrics 88
5.5.3 Performance of proposed Frag-LSTM-LID method 89
5.6 Conclusion 91

6 BI-DIRECTIONAL FRAGMENT NETWORKS FOR DATING HISTORICAL MANUSCRIPTS 92

6.1 Introduction 92
6.2 Problem formulation 94
6.3 BiD-FragNet for historical manuscript dating 95
6.3.1 Fragmentation (FR) and defragmentation (DF) 96
6.3.2 Main pathway 97
6.3.3 Fragment pathway 98
6.4 Results and Discussion 100
6.4.1 Evaluation metrics for manuscript dating 101
6.4.2 Performance of various techniques with MPS dataset [77] 101
6.5 Conclusion 104

7 FUNNELLING ENSEMBLE METHOD FOR WRITER IDENTIFICATION (FEM-WI) FOR HISTORICAL MANUSCRIPTS 105

7.1 Introduction 105
7.2 Handwritten writer identification (HWI) 106
7.3 Problem statement 108
7.4 Funnelling ensemble method for writer identification (FEM-WI) 108
7.4.1 Raw feature extraction 109
7.4.2 Base classifier training 109
7.4.3 Meta-feature computation 110
7.4.4 Meta-classifier training 110
7.5 Proposed features for writer identification 110
7.5.1 Histogram of orientation (HOO) pdf (*f1*) 111
7.5.2 Fraglet histogram of oriented displacement (FHOD) (*f2*) 111
7.5.3 Relative run length (RRL) (*f3*) 112
7.5.4 Zoned relative run length (ZRRL) (*f4*) 115

7.5.5 Contour hinge (CH) pdf (*f5*) 116
7.5.6 Modified contour hinge (mod-CH) pdf (*f6*) 118
7.6 Evaluation method for proposed features 121
7.7 Funnelling ensemble method (FEM) 122
7.8 Getting output class of test word (w_t) 123
7.9 FEM and Stacking ensemble technique 124
7.10 Results and Discussion 125
7.10.1 Evaluation datasets 125
7.10.2 Evaluation metrics 126
7.10.3 Evaluation performances of proposed features 126
7.10.4 Performance of individual proposed features 126
7.10.5 Performance of combined proposed features 127
7.11 Performance of FEM-WI technique 128
7.12 Conclusion 130

8 CONCLUSION AND RECOMMENDATIONS 132
8.1 Summary of work done in this thesis 132
8.2 Contributions 134
8.3 Future scope 134

PUBLICATIONS 135

REFERENCES 137

List of Figures

2.1 Classification of binarization methods 10

3.1 Degraded DIBCO images (1^{st} column) with different forms of degradations and their respective binarized forms (2^{nd} column) (ground truth): (a) DIBCO 2009 image (background noise and see throughs) [118], (b) DIBCO 2017 image (uneven illumination)[120], (c) DIBCO 2013 image (artifacts) [119], and (d) DIBCO 2013 image (see throughs) [119] 23

3.2 Framework of MLP-based binarization for historical manuscript images . . . 24

3.3 Gray values (p_i) fuzzification using triangular membership functions where μ_1, μ_2, μ_3, and μ_4 are respectively means of clusters (fuzzy regions) 1-4 and $\mu(p_i)$ is membership grade for pixel p_i. The broken lines denote cluster boundaries. 26

3.4 Binarization outputs for proposed method. Column (a) are degraded images from DIBCO 2013 and H-DIBCO 2014 [119, 127] and their binarized outputs in column (b) 32

3.5 Comparison of binarization results; (a) Degraded image [124], (b) Otsu [6], (c) Sauvola and Pietikainen [11], (d) Niblack [8], (e) Singh et al [13], (f) Bernsen [14], (g) Lu et al [12], (h) Mitianoudis and Papamarkos [18], (i) Bhowmik et al [136], and (j) Proposed MLP-based method. 33

4.1 Core word segmentation of handwritten words with overlapping and crossings: (a) Crossing words in (i & iii), (b) Crossing words (i & ii) and touching words (ii & v), (c) Over-segmentation (i) and under-segmentation (ii) of crossing words in e, (d) overlapping words, (e) crossing words, and (f) full segmentation of crossing words in e achieved by the proposed CTA technique. 42

4.2 Framework for CTA technique for word segmentation where the input is a binarized handwritten word image. 43

4.3 (a)Line segmentation of a vertical stripe of handwritten document (HWD) where: a(i) is vertical stripe of HWD, a(ii) raw (full/black line) and smoothed (dashed/blue line) horizontal projection profiles of vertical stripe in a(i), a(iii) is integral of smoothed profile in a(ii) showing up-turn point (L_u), local maxima point (L_p), and down-turn point (L_d), and a(iv) is a vertical stripe with line segmentations. (b) is handwritten document showing line boundaries. . 46

4.4 Handwritten adjacent connected components "got" and "ha" showing convex hulls (dashed/blue lines), bounding box (full/black lines), principal lines, hull distance (d_h), principal hull distance(d_{ph}), and bound box distance (d_b). . . 47

4.5 (a) A pair of words with crossing strokes at a junction point circled, (b) junction point (J) shown for crossing words in (a) consisting of 4 junction branches: XJ, JY, AJ, and JB, (c) Crossing strokes showing a tangent (T_1T_2) to AJ at J and mirror line (M_1M_2) for AJ 50

4.6 Framework of CTA technique for full word segmentation without under/over-segmentation. The inputs are CSW_i and thinned HWD image (I_{th}) while the output is full segmented word 53

4.7 Handwritten word spotting (HWS) framework 55

4.8 Local centered-slant correction for a foreground pixel Pi where θ_s is slant angle and θ_{ds} is de-slanting angle. 57

4.9 Normalization of word images by local centered-slant correction method. Left column shows skewed word images and right column are their respective normalized and binarized word images 58

4.10 Framework of IHOD shape descriptor 59

4.11 CHOD computation: (a) is a thinned word image $I_t(x, y)$ divided into $m \times m$ cells, and (b) is a $m \times m$ cell showing foreground pixels *p1* and *p2*. d_1 and d_2 are, respectively, displacements of foreground pixels p1 and p2 from cell center $C(x, y)$ whereas θ_1 and θ_2 are respectively orientations of *p1* and *p2* w.r.t $C(x, y)$ 60

4.12 Fully connected MLP network architecture used to develop handwritten word spotting model, where k is number of output classes and n is input feature size 64

4.13 Handwritten word segmentation results by CTA technique 67

4.14 Performance of CTA segmentation technique for non-LLT and LLT categories for (a) ICDAR2009 [155, 156], and (b) ICDAR2013 [157] datasets 68

4.15 Word segmentation results by CTA word segmentation method for crossing words, (a) sample crossing words segmented, and (b) graphical representation of segmentation performance of CTA word segmentation method 70

4.16 Sample instances of augmented word images in columns 1-4 (a-d) obtained by applying affine transformation and elastic distortion [168] on respective original images in columns 0 (a-d). Original image in row (a) is from RoyB dataset [39] whereas for rows b and c, original images are from IIIT-HW-DEV dataset [167] 72

4.17 MLP-based word spotting model performance plot for various cell sizes used during IHOD feature extraction, where kPr is k-precision, MAP is mean average precision, and m is cell dimension in pixels 74

4.18 Retrieved word images of IIIT-HW-DEV dataset [167], using MLP-based HWS technique where column 0 consists of query images and columns $1 - 5$ has retrieved instances of words in descending order of scores 75

4.19 Precision of the MLP-based word spotting technique on k top words retrieved from RoyDB [39] and IIIT-HW-DEV [167] datasets 76

4.20 MAP of the MLP-based word spotting technique on k top words retrieved from RoyDB [39] and IIIT-HW-DEV [167] datasets 77

5.1 Framework of Frag-LSTM-LID where FragWord denotes fragmented word or sub-word, PP means pre-processing, LSTM means long short-term memory, and f_i denotes outputs of LSTM networks 81

5.2 Architecture of LSTM cell where x_t is current input data point, c_t is new updated memory (cell state), c_{t-1} is memory (cell state) from previous LSTM unit, h_t is current LSTM output (or current hidden state), h_{t-1} is output (hidden state) of previous LSTM unit, f_t is output of sigmoid layer, f_{go} is forget gate output, C_u is useful information to be stored in current cell state, σ is sigmoid layer, tanh is tanh layer, $\oplus$ is adding information, and $\otimes$ is scaling of information 84

6.1 Evolving over time (horizontally) of handwriting styles for letters a, d, g, and p respectively from top downwards [62] 93

6.2 Framework of BiD-FragNet consisting of main and fragment pathways, where © denotes convolution for i^{th} level, G_i denotes feature maps of i^{th} level convolution layers in main pathway, D_i denotes feature maps of i^{th} level convolution layers in fragment pathway, FR denotes fragmentation of G_i to obtain A_i, DF denotes defragmentation of D_i to obtain B_i, M_i denotes result of i^{th} level concatenation of G_i and B_i, F_i denotes result of i^{th} level concatenation of D_i and A_i, and $\bigoplus$ denotes concatenation 95

6.3 Fragmentation and defragmentation in i^{th} level in BiD-FragNet network where G_i is feature map of convolution layer in main pathway, B_i is defragmented feature map obtained by joining together feature maps D_i. D_i are feature maps of convolution layer in fragment pathway, and A_i are fragment feature maps resulting from fragmentation of G_i 97

6.4 Cumulative score for hinge [44], quill [45], junclets [71], $SF + CF$ [75], quill-hinge [45], delta-hinge [193], strokelets [76], polar stroke descriptor (PSD) [75] features, and the proposed BiD-FragNet for various error values for MPS data set [77] 103

7.1 Framework of FEM-WI system where f_n^1 is n^{th} input/base feature descriptor, h_n^1 is n^{th} base classifier, and f_n^2 is n^{th} meta-feature descriptor obtained as output of h_n^1 and $n = 1, 2, ...N$ is the index of base/meta feature/classifier with N being number of features used, and h^2 is meta classifier. 108

7.2 Computing histogram of orientation (HOO) and Fraglet histogram of oriented displacement (FHOD) features for fraglet AB, where P_i is fraglet pixel point, P_c is center of fraglet's bounding box, θ is orientation angle of P_i w.r.t P_c, and d_i is displacement of P_i from P_c 112

7.3 Categories of white runs for a binary word image (god) with foreground represented by 0 (black) and background region represented by 1 (white) where (a) shows horizontal white runs for 4 selected rows, and (b) shows vertical white runs for 4 selected columns. Dashed lines represent bounding box boundary lines 113

7.4 Contour hinge formed from 2 fraglets (hinge legs) AB and BC joined at a hinge-point B: (a) illustration of angles θ_1 and θ_2 in modified-CH where BD is bisector of angle ABC on concave side of contour hinge ABC, and (b) illustration of angles ϕ_1 and ϕ_2 in contour hinge pdf [44] 117

7.5 Contour of handwritten letter with hinges ABC and DEF 118

7.6 Handwritten text of 2 writers in a(i) and b(i) where a(ii) and b(ii) are their respective graphical representations of CH distributions. a(iii) and b(iii) are graphical representations of mod-CH distributions respectively for a(i) and b(i) texts 120

List of Tables

3.1 Performance of MLP-based binarization method with (H)DIBCO 2011-2013 test datasets [119, 124, 126] 32

3.2 Comparison of performance of the proposed binarization method with state-of-the-art methods on DIBCO 2011 printed document images [124] 34

3.3 Comparison of performance of the proposed binarization method with state-of-the-art methods on HDIBCO 2012 handwritten document images [126] . . 35

3.4 Comparison of performance of the proposed binarization method with state-of-the-art methods on DIBCO 2013 handwritten document images [119] . . . 35

3.5 Comparison of performance of the proposed binarization method with state-of-the-art methods on DIBCO 2013 printed document images [119] 36

4.1 Comparison of performance of CTA segmentation method with state-of-the-art segmentation methods for ICDAR2009 [155, 156] and ICDAR2013 [157] . 69

4.2 Performance of CTA word segmentation method in segmenting crossing words 70

4.3 Performance of MLP-based word spotting technique for various cell dimensions ($m \times m$) used during IHOD feature extraction 74

4.4 Comparison of performance of HWS method with state-of-the-art word spotting techniques for RoyDB [39] and IIIT-HW-Dev [167] datasets 76

5.1 Hyperparameters of LSTM networks 87

5.2 Hyperparameters for dense (fully connected) layers 87

5.3 Performance of Frag-LSTM-LID technique in language identification with UDHR [183, 184, 185] and Kenya indigenous languages (KIL) datasets . . . 90

5.4 Confusion matrix for KIL corpus 91

6.1 Comparative performance evaluation of BiD-FragNet dating method for MPS data set [77] 102

6.2 Confusion matrix for performance evaluation of the proposed dating method for MPS data set[77] 104

7.1 Expressions for RRL for various WRs from $H \times W$ word image in figure 7.3 115

7.2 Performance of proposed features with IAM[207] and FireMaker[166] datasets with NNM 127

7.3 Performance of combined features with IAM[207] and FireMaker[166] datasets 128

7.4 Performance of the proposed FEM-WI method with various features for IAM[207] and FireMaker[166] datasets 130

7.5 Performance comparison of FEM-WI method for aggregated meta-classifier outputs with state-of-the-art writer identification methods for IAM [207] and FireMaker [166] datasets 131

Chapter 1

INTRODUCTION

1.1 Need for historical manuscript studies and associated challenges

Historical manuscripts are handwritten documents from historical periods. They are valuable sources of knowledge regarding social events, notable persons, origin and heritage of people, language use, culture, governments, organisations, and societies [1, 2]. A physical historical manuscript consists of 2 main parts: textual content and writing support [3]. Text content is the written content or information that is in form of characters, words, inscriptions, or graphics. Writing support is a physical object whose surface bears the text, inscription of graphical content.

There are 5 main materials used as writing support: papyrus, parchment, vellum, cloth, and paper. Papyrus is made by joining together thin dry strips of papyrus plant in a rectangular shape using an adhesive. A scroll is formed by joining many such rectangular pieces. The scroll is then written on one side. Parchment is made of skins of animals like goats, sheep, and antelopes. It is of better quality than papyrus and was used between $2 - 15^{th}$ centuries [4]. Vellum is similar to parchment, but the difference is that skins of young animals (like kids, lambs, and calves) are used to make it. Its quality and cost are higher than that of parchment [4]. Cloth writing material was used on rare cases. Paper is a more-modern

writing material invented about 2^{nd} century. It is made from wood or trees. Paper was not commonly used as a writing material during historical eras [5].

The common tasks involved in studying, processing and managing historical manuscripts include: (i) restoration of degraded manuscripts, (ii) indexing and classification, (iii) manuscript retrieval, (iv) language identification of manuscript text, (v) author/writer identification, and (vi) Era or production time prediction. Historical manuscripts have a rich history about the past, thus, their study enriches modern people with insight about the past. However, in the course of their study and management, a number of challenges abound as stated here below:

(a) Due to their historical origin, most of the historical manuscripts suffer from degradations like ageing, wear and tear, fading of textual content, and ink bleed among others. The degradations are caused by environmental factors, poor handling and storage mechanisms.

(b) Some approaches of studying, processing, and managing historical manuscripts need actual historical manuscripts which in turn hasten degradation and destruction of manuscripts.

(c) Some methods for studying manuscripts are costly especially those using specialized and expensive instruments like carbon dating and spectroscopic equipments for determining age of manuscripts.

(d) Other methods are manual and human-based which are subjective and slow. Thus, they are error prone, tiresome, and unfeasible in large scale.

1.2 Computer-based management of historical manuscripts and its issues

Due to availability and advancement of information and communication technology (ICT) tools, many historical manuscripts have been digitized and to electronic forms as digital

images. The shift to scanned historical manuscript document images enhances sharing, research, and preservation of historical manuscripts. Computer-based methods (CBM) for management of historical manuscripts are necessary since they work on scanned manuscript documents, thus preserving them. These methods are employed in virtually all historical manuscript management tasks like restoration/binarization, classification, indexing, searches, dating, and sharing across online platforms.

Due to degradations of historical manuscripts, image processing techniques are applied for their restoration to former state with improved visual appearance. These approaches restore lost visual qualities of manuscript images. Machine learning (ML) methods are applied in classification of manuscript document images. The ML methods used include CNN (convolutional neural network), MLP (multi-layer perceptron), and SVM (support vector machine) among others. Also, MLP and SVM have been used in dating manuscript images. The need for computer-based methods stems from the fact that they are objective, robust, efficient, and have high performance. These methods are feasible in large scale compared to manual and human-based methods.

1.3 Problem statement

There is continual research geared towards studying and management of historical manuscripts in tasks like restoration, indexing, classification, manuscript search, and dating. The main objective of this thesis is to develop an efficient computer-based framework to classify historical manuscripts. The specific problems addressed in this thesis are as follows:

(i) Most historical manuscript images suffer from different forms of degradation like ink bleed, fading, artifacts, see throughs, uneven illumination which decrease their quality. These degradations are a challenge to many heuristic-based approaches for binarization. There is background noise in binary outputs.

(ii) Word spotting approach is used in indexing, content-based retrievals and searching in historical manuscripts. Most of the existing word spotting techniques employ features

which have low discriminating power, resulting to low output performance. Some methods have high computational cost like CNN-based techniques.

(iii) Most of word segmentation techniques perform well in good quality handwritten manuscripts with well-spaced words and sentences but underperform in cases where word/sentences are overlapping. Their outputs are under/over-segmented words which in turn will affect performance of the systems/steps using such words with segmentation errors.

(iv) Writer identification being a major manuscript management task, many non-model-based techniques are already in the literature. These techniques use features that have low discriminating powers, thus cannot characterize and distinguish different authors well. These methods are also text dependent hence unsuitable in variety of language scripts.

(v) Since features are used in dating of historical manuscripts, they should have low variance within same time unit and large variance between different time units. Most of the features used are the same ones used for writer identification which discriminate well between different writers but are poor in capturing trend of handwriting style over time. This challenge is common in hand-crafted features.

(vi) A majority of language identification methods do not perform well when tested on large number of languages. Some are not able to distinguish languages that are semantically related since they use features that cause such a limitation.

(vii) The existing classification approaches are limited to a few criteria. An elaborate manuscript classification ought to employ many classification fields like language, age or date of creation, layout, author, subject content, geographical origin, among others. Other methods are not efficient and robust enough in large scale since handcrafted features are used instead of learned features.

1.4 Scope and objectives of the thesis

Computer-based approaches like computer vision and machine learning (ML) approaches are required in various fields. Machine vision finds use in various applications like manuscript management, face recognition, criminal investigation tasks, vegetation study, medical diagnosis among others. Some of them are briefly discussed below:

- Manuscript study. In studying historical manuscripts, various tasks carried out include classification, author identification, and layout structure analysis. These tasks are based on explicit and implicit attributes of manuscripts. Computer vision is often employed in extracting the attributes and automating the stated tasks.
- Manuscript restoration. Computer vision applicationsML, ML, and image processing are employed in improving visual appearance of manuscripts by removing their degradations of different forms. Degradations are caused during scanning/digitization phase, poor storage conditions, and poor handling mechanisms.
- Document image analysis. These days scientists are using computer-based applications for document analysis. Out of these, handwriting feature of different authors, language structure analyses are gleaned from historical manuscripts and used by criminal investigation agencies, security authorities, and historical manuscript archiving centres.

The following are the objectives of the thesis:

- ❖ For every manuscript document image, it should first be binarized as a primary preprocessing step for subsequent steps. Thus, a binarization technique is required that is efficient in the presence of degradations of different forms.
- ❖ In indexing manuscript document images, word spotting approach is used. Efficient word spotting methods are needed especially when working with handwritten manuscript documents where there is overlappingand crossing of words.

- ❖ Some of the major challenges associated with textual contents of historical manuscripts is overlapping and crossing of words and non-uniform base lines. These factors pose a challenge to word segmentation, and author identification process. Therefore, there is need to develop word segmentation and author identification techniques that are robust in the presence of these challenges.

- ❖ Language identification and era or production time estimation are prime attributes of historical manuscripts that are challenging to undertake. These tasks need to be undertaken since they are used in understanding and classification of manuscripts.

1.5 Contribution and Thesis Layout

The problems mentioned in section 1.3 informs our motivation to come up with their solutions which are presented in detail in this thesis. In this thesis, effective solutions have been put forward to address the challenges of binarization of poor quality and degraded manuscript document images, word spotting for indexing handwritten manuscript images, handwritten word segmentation, writer identification, dating and classification of historical manuscripts. The main contributions in this thesis are as follows:

(i) A model-based technique has been developed for enhancement of historical manuscript images by binarization approach. Handcrafted features with high discriminating powers have been extracted from manuscript images which are in turn used to develop a binarization model via training multi-layer perceptron (MLP).

(ii) Words are inputs for most document analysis systems. Thus, word segmentation is a crucial primary step in such systems. Therefore, a novel component tracing and association (CTA) word segmentation technique has been proposed that is devoid of under/over-segmentation especially in cases of overlapping and crossing word images.

(iii) Key word spotting approach has been used so as to achieve high performance in word image retrieval/search from historical manuscript document images. In carrying out this task, Integral Histogram of Oriented Displacement (IHOD) shape descriptor has

been developed. IHOD attained very good performance since it has high discriminating power. This descriptor is also used for indexing for historical manuscript images.

(iv) Handwriting styles varies between and within writers for the same word instance. Features capturing handwriting styles of different writers are used for writer identification task. To achieve this task (writer identification), a novel offline and text-independent model-based technique called funnelling ensemble method for writer identification (FEM-WI) is proposed. FEM-WI is a bi-level system of MLP classifiers where different features are used to train level 1 classifiers. Level-2 classifier is then trained by outputs of level-1 classifiers. Level-2 classifier is used to give final output.

(v) (v) A novel deep learning-based technique called Bi-Directional Fragment Networks (BiD-FragNet) has been developed for predicting era or creation date of historical manuscript images. BiD-FragNet is composed of 2 CNN-based pathways, i.e., main and fragment pathways. These two pathways are used to extract global and local features. The pathways share information at some levels thus making the extracted features to have high discriminating powers.

(vi) In order to address the challenge of language identification, Fragmented long short-term memory (LSTM) technique has been proposed. In the proposed technique, local and global features are extracted from fragmented and whole textual word respectively using LSTM architecture. The features are then used to develop an efficient language identification model.

Thesis layout is as follows:

Chapter 1: Introduction

Chapter 1 discusses nature and significance of historical manuscripts along with various management tasks and processes associated with them. Contributions and thesis layout is also discussed.

Chapter 2: Literature Review

Chapter 2 discusses related works done covering approaches used in various tasks and processes pertaining to manuscript management. The approaches focus on manuscript binariza-

tion, word spotting, writer identification, language identification, dating, and archiving and classification.

Chapter 3: Manuscript enhancement by binarization

This chapter discusses enhancement of degraded historical manuscript images using binarization technique.

Chapter 4: Handwritten word spotting for indexing historical manuscripts

This chapter discusses indexing of historical manuscripts using key-word spotting method that uses integral histogram of oriented displacement (IHOD) descriptor.

Chapter 5: Language identification for historical manuscript documents

The chapter presents a technique using fragmented LSTM (long short term memory) to identify language of texts of historical manuscripts. The method utilizes word embeddings of fragmented or subdivided words.

Chapter 6: Bi-directional fragment networks (BiD-FragNet) for dating historical manuscripts

This chapter discusses a deep learning-based technique for estimating era or creation date of historical manuscript images. Two CNN-based pathways are used to learn and extract local and global features from historical manuscript images that are then used predict their era or production date.

Chapter 7: Funnelling ensemble method for writer identification (FEM-WI) for historical manuscripts

Chapter 7 discusses a model-based Funnelling Ensemble Method that is applied for identifying authors of historical manuscript images.

Chapter 8: Conclusion and recommendations

In this chapter, findings of the proposed work are presented in summary. Also, future work is presented.

Chapter 2

LITERATURE REVIEW

2.1 Introduction

The tasks associated with study and management of historical manuscript images include restoration, enhancement, word spotting, writer identification, manuscript dating, language identification, and manuscript classification. For the last 4 decades, researchers have been working on these tasks/fields using various techniques. Since most of historical manuscript images suffer from degradation of different forms, many of the methods put forward to handle the afore-mentioned tasks depend on degradation status of the manuscripts. The methods work with digital scans of historical manuscripts.

2.2 Restoration methods for degraded historical manuscript images

Binarization is one of the ways of restoring degraded historical manuscript images. Binarization is a process of representing image pixels in two levels only, where one level is for background pixels and another for foreground pixels. Binarization techniques can be divided to 2 broad groups: heuristic and model-based methods. Heuristic methods mainly use statistical approaches to separate background and foreground regions of a manuscript image.

Heuristic methods are further sub-divided into threshold-based, non-threshold-based, and hybrid methods. Model-based methods mainly used statistical learning approaches to develop a model for separating background and foreground regions of an image. The methods are further sub-divided based on the features used to develop a binarization model: deep learning-based and handcrafted features-based methods. Figure 2.1 shows a conceptual view of classification of binarization methods. The methods are discussed in sections 2.2.1 – 2.2.5.

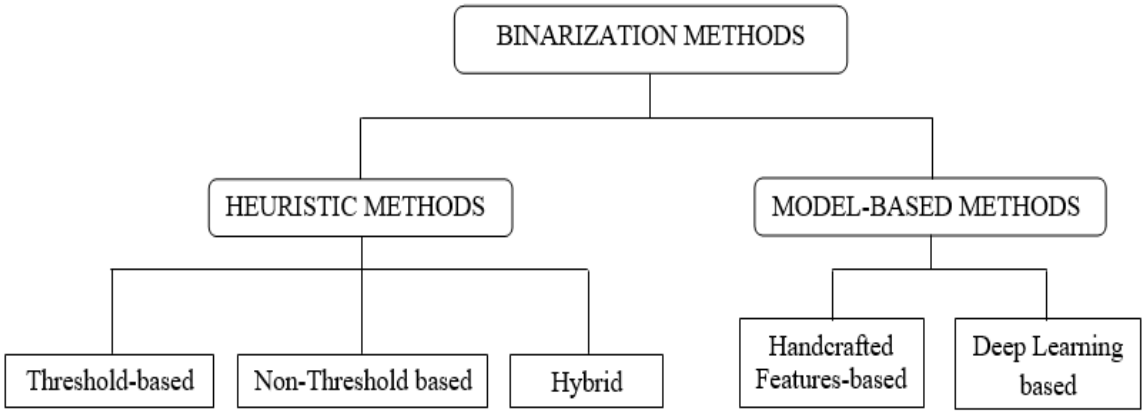


Figure 2.1: Classification of binarization methods

2.2.1 Threshold-based binarization methods

These methods use threshold pixel value as a boundary separating foreground and background pixels. A pixel with a value greater than threshold value is foreground pixel whereas a pixel whose value is less than threshold value is a background pixel. Threshold-based methods are further divided to local (adaptive) and global techniques. Global techniques use one threshold value for whole image whereas local/adaptive techniques use different threshold values for different regions in an image. Global methods include Otsu method [6] where global threshold is computed in such a way to minimize intra-class variance. Local methods include Xueting et al. [7] and Niblack [8]. In method by Niblack [8], global mean pixel and local variance are used to compute local threshold. The Niblack method [8] was improved by Jayanthi and Indu [9] and Khurshid et al [10]. However, both methods suffer from noise in background regions. Other local methods include Sauvola and Pietikainen [11] who used windows of varying sizes to obtain local thresholds, Lu et al [12], Singh et al [13], and Bernsen

[14] who used local contrast and local mean pixel to obtain local threshold value for binarization, Gatos et al [15] who used background surface and pre-processed degraded image to compute local threshold. Local methods outperform global methods especially in degraded images. Global methods perform well in good quality images with uniform illumination but fail in images with non-uniform illumination.

2.2.2 Non-threshold-based binarization methods

These methods do not employ threshold values when binarizing document images. Vinod and Niranjan [16] used wavelet decomposition approach on document images where coefficients obtained were used for binarization. Papamarkos [17] used neuro-fuzzy technique to binarize document images. Mitianoudis and Papamarkos [18] used Gaussian mixture modelling (GMM) method for binarization. Howe [19] used Laplacian energy of gray values to binarize document images. These methods are computationally expensive as compared to threshold-based methods.

2.2.3 Hybrid binarization methods for historical manuscript images

These methods integrate both local and global methods to binarize images. They include methods by Wu and Amin [20] and Tanaka [21].

2.2.4 Handcrafted features-based binarization methods

These methods model foreground and background regions of a document by training a suitable classifier using handcrafted features. The features used are manually extracted from an image using designed algorithms. Example of this method is that of Kefali et al [22] who used local and global information of images to train multi-layer perceptron (MLP) classifier for binarization of old manuscripts. Targets used are ground truth images. Wu et al [23] used statistical features associated with pixels to develop a binarization model for binarizing

document images. The features used include local mean, exponential truncated Niblack index, pixel intensity deviation from Otsu's threshold, contrast, Logarithm Intensity Percentile (LIP), global mean, Logistic Truncated Sauvola Index, (LIP) features, global standard deviation, and Relative Darkness Index features. Another model-based binarization method using handcrafted features is that of He and Schomaker [24].

2.2.5 Deep learning-based binarization methods

They use learned features to develop binarization models. The features are extracted by neural networks like LSTM (long short-term memory), and CNN (convolution neural network). An example of such methods is Afzal et al [25] who used deep-learning approach for binarization of document images where 2D LSTM was used to classify pixels to background and foreground categories. Tensmeyer and Martinez [26] employed FCN (fully convolutional network) to binarize document images. In this method, features are learnt and extracted by convolutional layers at 4 scales (levels). Also, CNN-based methods have been used by Peng et al [27] and Akbari et al [28] to binarize document images.

2.3 Word spotting techniques for historical methods

Word spotting methods can be divided to three main categories: (i) segmentation-based methods, (ii) segmentation free methods, and (iii) learning-based methods. Learning-based word spotting techniques use models obtained by training a suitable classifier using features extracted from historical manuscript images. The classifiers used include HMM (hidden markov model) [29], neural networks [30], and SVM (support vector machine) [31]. Segmentation-based word spotting methods utilize segmented words whereas segmentation free methods do not use segmented words/sentences from manuscript documents. Learning-based methods have better performance than non-learning-based techniques [32]. Bhaldwaj et al [33] extracted moment-based features from segmented words and used them for word spotting. Cosine similarity scores were used to compare query and candidate words so as to obtain top-ranking word. Srihari et al [34] has successifuly used word shape method for key

word spotting in Arabic and Devanagari document images. Shekhar and Jawahar [35] have used the same approach as Srihari et al [34] for word spotting in Bangla scripts, where the features used are bag of visual words (BoVW) and scale invariant feature transform (SIFT). Hassan et al [36] used word shape descriptor-based technique for word spotting for Bangla and Devanagari scripts for purpose of indexing document images. Best precision of 88% was achieved. Sudholt and Fink [30] used convolution neural network (CNN) to develop a word spotting system, where training was done using a representation of pyramidal histogram of characters. Zhang et al [37] carried out word spotting for Bangla handwritten document images using segmentation free method. Heat Kernel signature (HKS) features were used to represent key points of word images. Zagoris et al [38] used key points based on local gradient vectors and orientations extracted from words as features for word spotting. In this method, variations among similar words due to slanting and varying character sizes are not considered. Roy et al [39] used zonal features to perform word spotting in document images written in Indic languages like Devanagari and Bangla. The features are obtained from horizontal divisions/layers (zones) of a word image. Zonal features are matched to find other instances similar to query word. Another work where zonal-wise approach has been used for word spotting is that of Bhunia et al [40]. Roy et al [39] has reported that zoning-based word spotting methods have higher performance than word spotting where full word images are used.

2.4 Auto-Writer identification methods for historical manuscript images

Writer identification methods can be grouped to 2 broad classes: Deep learning-based (DLB) and handcrafted features-based (HFB) techniques.

2.4.1 Handcrafted features-based writer identification methods

These methods use features manually extracted from historical manuscript document images using designed algorithms [41, 42]. In most cases, HFB-writer identification methods do not use models for writer identification tasks. They employ nearest neighbour search approach. This approach uses computed distance between query word image and all candidate word images in the database with known authorship. The author of the candidate word image with least distance is deemed to be the author of the query word.

Grapheme shape complexity (GSC) descriptor has been used by Bensefia and Djeddi [41] to describe handwriting style for writer identification. GSC captures graphical complexity of segmented graphemes, computing them using Fourier elliptic transform [43] of closed contours of binarized graphemes. Bulacu and Schomaker [44] used contour-based texture features and allographs to develop a text-independent writer identification technique for characterizing handwriting styles of authors. They also combined features that improved performance by more than 5%. Quill features have been used by Brink et al [45] for writer identification. The features are computed from ink trace width and direction to characterize handwriting styles of individual authors. Since ink trace width is determined by writing surface and writing instrument, efficiency of quill feature is affected in cases where different writing instruments and writing surfaces are used by same author. Lai and Jin [46] used local path signature (LPS) feature for writer identification. Chahi et al [47] used local convex micro-structure patterns (LCxMSP) features for writer identification. LCxMSP are local patterns built in 3×3 neighbourhood of a pixel which captures handwriting styles of an author. Another handcrafted feature used for writer identification tasks is Local binary pattern (LBP) [48, 49, 50].

2.4.2 Deep learning-based (DLB) writer identification methods

In these methods, features used are learnt and extracted from historical manuscript images using deep neural networks (DNN) like CNN [2, 51]. DLB methods have better performance compared to HFB writer identification methods.

He and Schomaker [52] used fragment network (FragNet) method to identify text authors based on learning features via two-pathway convNet (feature pyramid and fragment pathways). Both feature pyramid and fragment pathways consist of CNN architectures where fragment pathway processes word image fragments whereas feature pyramid pathway processes full word images. Feature pyramid pathway shares its information with fragment pathway at some set levels. Feature pyramid pathway extracts global features while fragment pathway extracts localised features. The combined outputs of feature pyramid and fragment pathways are used as word features to characterize handwriting styles of various authors. Xing and Qiao [53] presented a 2-stream (pathway) CNN-based technique for writer identification. During training, same image patch is concurrently input to the 2 channels which share information in between. The outputs from last fully connected layers of both pathways are summed before being fed to softmax for final classification. Cilia et al [54] employed a 3-step deep neural network (DNN)-based system for Medieval writer identification. The steps are: (i) text line/row detection from medieval manuscript, (ii) feature extraction from text line and its classification, and (iii) weighted voting to get final decision on writer of a given manuscript page. Voting is done using line classification results. Nguyen et al [55] have also used deep learning-based end-to-end method for writer identification.

2.5 Historical manuscript dating (HMD) techniques

HMD techniques can be divided to 3 broad groups: physical, paleographical, and computer-based techniques.

2.5.1 Physical techniques for HMD

In these methods, creation date or age or era of historical manuscript documents are determined by measurement and analysis of constituent components/elements of actual sample of historical manuscript document. The components analysed can be of ink or writing material/support. The most common methods include carbon-dating [56, 57] and spectroscopic techniques like Infrared (IR) spectroscopy [3, 58] and X-ray spectroscopy [59]. Radiocarbon

dating technique is used to determine age of carbon-based materials only. In carbonaceous materials, their radiocarbon (^{14}C) component decays with time in a process called radioactive decay. The rate of decay is constant throughout the life time and it varies for different carbon-based materials. For a given carbon-based writing material or ink, its amount of radiocarbon (^{14}C) content is measured and analysed to obtain its age or production date [60, 61]. This method was used to date Dead Sea Scrolls (DSS) [57]. Spectroscopic methods for HMD employ the interaction of electromagnetic (EM) radiation with matter that result in emission of spectra of frequencies which are characteristic for different writing materials. This helps to identify constituent elements of a given sample of historical manuscript document, which are further analysed to give age or production date. Physical techniques for HMD are destructive since actual samples are used. They are also costly and time consuming.

2.5.2 Paleographical methods for HMD

Paleographers are expert persons in studying handwritten historical documents. In HMD methods, paleographers use their expertise, intuition, experience, and skill to determine age or creation time of manuscripts. The paleographic process involves extraction of manuscript's visible or meta-data attributes and inference. Attributes extracted include language structure, manuscript layout, scribal abbreviations of words, ligatures, writing styles, character structure, punctuations, writing customs, and writing material used. These attributes are then used to infer era or creation time of a historical manuscript document. Comparison approach is also employed where creation-time of undated manuscript is estimated by comparing it with a similar dated/dateable manuscript [62, 63, 65, 66]. This approach is time consuming, error prone, and subjective since different manuscript experts may obtain conflicting eras or production times for same historical manuscript [62, 63]. A classic example is that of ancient Japanese calligraphies in which their production date as estimated by Paleographers is 12^{th} century AD. This production date conflicts with 14^{th} or 15^{th} century AD given with radiocarbon dating technique [56]. Another example is the old Hebrew Torah Scroll (HTS) that consists of five complete books of Moses of the old testament of the Holy Bible. Using paleographic process for HMD, an Italian librarian erroneously estimated pro-

duction date of this HTS to be 17^{th} century AD. This production date estimation took place in 1889 AD. However, reliable and correct production date of HTS was found to be between 1155-1225 AD upon using carbon-14 dating technique [64].

Mani and Wilson [65] and Llidó et al [67]) used frequency of time-tagged words in historical manuscript documents as metadata to determine their eras or creation times. Using palaeographical metadata, Paleographers have estimated production date of old Japanese calligraphies to be 12^{th} century AD [56]. Also, based on palaeographical information, production date of Papyrus-46 (P^{46}) manuscript has been determined by Kim [66] to be 1^{st} century AD. This same manuscript (P^{46}) has been dated to 3^{rd} century by Sanders [68]. Papyrus-46 (P46) is a New Testament manuscript (of Christian Bible) that is written in Greek language using papyrus material.

2.5.3 Computer-based techniques of HMD

These techniques employ computer-based approaches to predict creation time or age or era of historical manuscripts. These approaches use features that are extracted from the manuscripts using computer-based algorithms. Computer-based techniques for HMD can be split into 2 subclasses: image-based and semantic-based techniques. In **semantic-based HMD**, semantic attributes are used in capturing trend of usage of words and language over time. These semantic features are used in Statistical Language Modelling (SLM). SLM is used in dating manuscripts since it captures temporal information of a language w.r.t trend of usage of words and language over time. As a result, probabilities assigned to text, word, or phrases (due to frequency of their use over time) are used to obtain age or era or creation time of a historical manuscript.

In **image-based HMD** methods, hand writing style of authors is characterized by using features. The hand writing style is in turn used to estimate era or creation time of manuscripts. In these approaches, it is assumed that handwriting styles evolve over time. Therefore, the features capture temporal information on trend of evolving handwriting style over a period of time. Features used include ink-based features [45], bag of contour fragments (BCF) [69], discrete contour evolution (DCE) [70], polar stroke descriptor (PSD) [71], shape context

descriptors [72], and textural features [73]. Al-Aziz et al [74] used texture analysis to date Arabic historical manuscripts into 3 time periods; (i) contemporary or modern age (1220 – present), (ii) Ottoman age (923-1220), and (iii) Mamluk age (648-923). He et al [75] used handwritten pattern analysis to date undated historical manuscripts. The feature descriptor used for handwriting styles of the manuscripts are scale invariant mid-level PSD [76]. Stroke fragments and contour fragment features have been used by He et al [62] to date historical manuscripts. The fragment features describe writing styles of historical manuscripts. This dating technique rely on writer identification approach and gradual evolving and change of writing style over the period when the manuscripts were created. Hinge and Fraglets features have been used by He et al [77] for dating historical manuscripts from medieval paleographical scale (MPS) dataset. Hamid et al [78] proposed a dating system that used combined textural features of histogram of LBPs, Gabor filters, and uniform LBPs. Wahlberg et al [79] used deep learning approach to date Svenskt Diplomatariums Huvudkartotek (SDHK) dataset of historical manuscripts that consist of dated medieval charters [72, 80]. They tuned ImageNet network using SDHK dataset [72, 80] because of insufficient image samples in SDHK. Hamid et al [2] used pre-trained ConvNet in transfer learning for dating historical manuscripts from MPS dataset. The pre-trained ConvNet was used to extract features. The learned features are used with SVM for dating the manuscripts.

2.6 Language identification methods for historical manuscripts

Language identification (LID) tasks are basically model based, i.e., a language model is developed which in turn is used to identify language of a manuscript. Thus, LID methods are classified on the basis of approach used to develop a language model. In light of this, LID methods can be classified to three broad classes: word-based, N-gram based, and learning-based methods.

2.6.1 Word-based LID methods

In these methods, a language is modelled based on word frequency or word length. When word frequency is used, frequent words are used to represent a language model. When word length is used, words not exceeding a set length are used to represent a language model.

Souter et al [81], have used 100 most frequent words for LID with 9 languages: English, French, Spanish, Friesian, Portuguese, Gaelic, German, Italian, and Serbo-Croat. Accuracy of 91% was obtained. Dongen [82] and Kumar et al [83] have used word frequency as features for language identification with high performance. Other works where words frequency has been used as features for language identification include Clematide and Makarov [84], Saharia [85], Cagigós [86], and Gómez-Adorno et al [87].

2.6.2 N-gram-based LID methods

N-gram is a group of contiguous characters of a word. The size of N-gram ranges from $2-7$. To determine the language of a given script from N-Grams, out-of-place similarity measure method is used. N-gram frequency can be directly used as a feature for LID as used by Alrifai et al [88], Malmasi and Dras [89], Miura et al [90], Tellez et al [91], and Prager [92]. In other LID instances, discriminating N-grams for different languages are used as in the works of Bilcu and Astola [93], Murthy and Kumar [94], Babu and Baskaran [95], and Hayati [96]. Discriminating N-grams are those with sparse frequencies for different languages considered. They are selected using Fisher's discriminant and PCA (principal component analysis) (PCA) methods. Weighted N-grams have also been used by Malmasi et al [97] for LID. The N-grams are weighted using Term Frequency–Inverse Document Frequency (TF-IDF). In a departure from use of traditional N-grams, Saharia [85] used consonant bigrams and trigrams as LID features. These N-grams were obtained from words after removing vowels.

2.6.3 Learning-based LID methods

In these methods, language models are developed using learned features extracted from words using neural networks like LSTM, CNN, and recurrent neural networks (RNN). Cazamias et al [98], Hochreiter and Schmidhuber [99], and Samih et al [100] have employed LSTM for LID. RNN has been used by Bjerva [101], Jurgens et al [102] and Kocmi and Bojar [103] for LID. Various researchers have carried out LID task using CNN-based techniques with good resulys like Jaech et al [104], Jaech et al [105], and Li et al [106]. Also, HMM has been used for language identification [107, 108, 109].

Apart from the LID discussed so far, other features that are often used for LID include characters, morphemes, and phonemes. Morphemes have been used by Clematide and Makarov [84], Samih [110], and Darwish et al [111] for LID. Character level features that have been used for LID include vowel–consonant ratio features [82], character frequency/probability [112, 113, 114], and character capitalization [115, 116, 117]. Of all these approaches, learning-based approaches have the best performance since they model languages well.

Chapter 3

MANUSCRIPT ENHANCEMENT BY BINARIZATION METHOD

3.1 Introduction

This chapter presents a model-based binarization technique for enhancement of degraded historical manuscript images (HMI). The binarization model is obtained by training multi-layer perceptron (MLP) classifier, that then classifies HMI pixels to either background or foreground. HMIs suffer from different forms of degradations like fading, see-throughs, ink bleeds, poor and uneven illumination caused during scanning (digitization) process. These degradations are accrued due to ageing, environmental storage conditions like dampness, mishandling, maintenance/preservation processes, deliberate tampering, and natural quality decline among others. These degradations decrease quality of the document images, consequently compromising efficiency of any document analysis system using such images as input. Therefore, the quality of document images needs to be enhanced before being fed to a document analysis system for success in subsequent steps and increase of overall output performance of the system. Binarization is one of the main ways of enhancing manuscript document images. Binarization removes the afore-mentioned degradations and noise, and hence enhance performance of subsequent tasks and system's overall performance.

3.2 Binarization of HMI

Binarization is a process of representing image pixels in two levels only, where one level is for background pixels and another for foreground pixels. Foreground pixels represent the wanted region, also referred to as object region. Foreground region can be text, isolated character, diagram, figure, table or a drawing among others. Background region represent the unwanted pixels like noise or non-object pixels. In the case of HMIs, foreground region corresponds to object or text whereas background region corresponds to non-textual area that is normally plain region of the manuscript. Also, background region may be having unwanted pixels like noise or non-object pixels (like fading, ink-bleeds, see throughs, artifacts) which need to be removed during binarization process. Figure 3.1 shows document images from DIBCO 2009, 2013 and 2017 [118, 119, 120] with various forms of degradations and their respective binarized forms. The figure shows document images with background noise (figure 3.1a), see throughs (figures 3.1(a & d)), uneven illumination (figure 3.1b), and artifacts (figure 3.1c). Besides binarization being a primary step in manuscript analysis systems, it also enhances perceptual quality of a manuscript document.

In this chapter, multilayer perceptron (MLP)-based model is used to binarize historical manuscript images (HMI). In this method, low level fuzzy features and statistical features are obtained from degraded document images and used to build a binarization model by training MLP classifier by backpropagation.

3.3 Problem statement

Many of the heuristic approaches for binarization (like global and local-based methods) do not perform well especially in cases of severely degraded HMI as a result of various aforementioned forms of degradations. Training-based techniques help address these challenges. Low-level hand-crafted features are used to build statistical models for classifying pixels to background and foreground. This approach effectively handles image degradations where heuristic methods fail. In the proposed method, low level fuzzy and statistical features are obtained from degraded documented images (section 3.4.2) and used to build a binarization

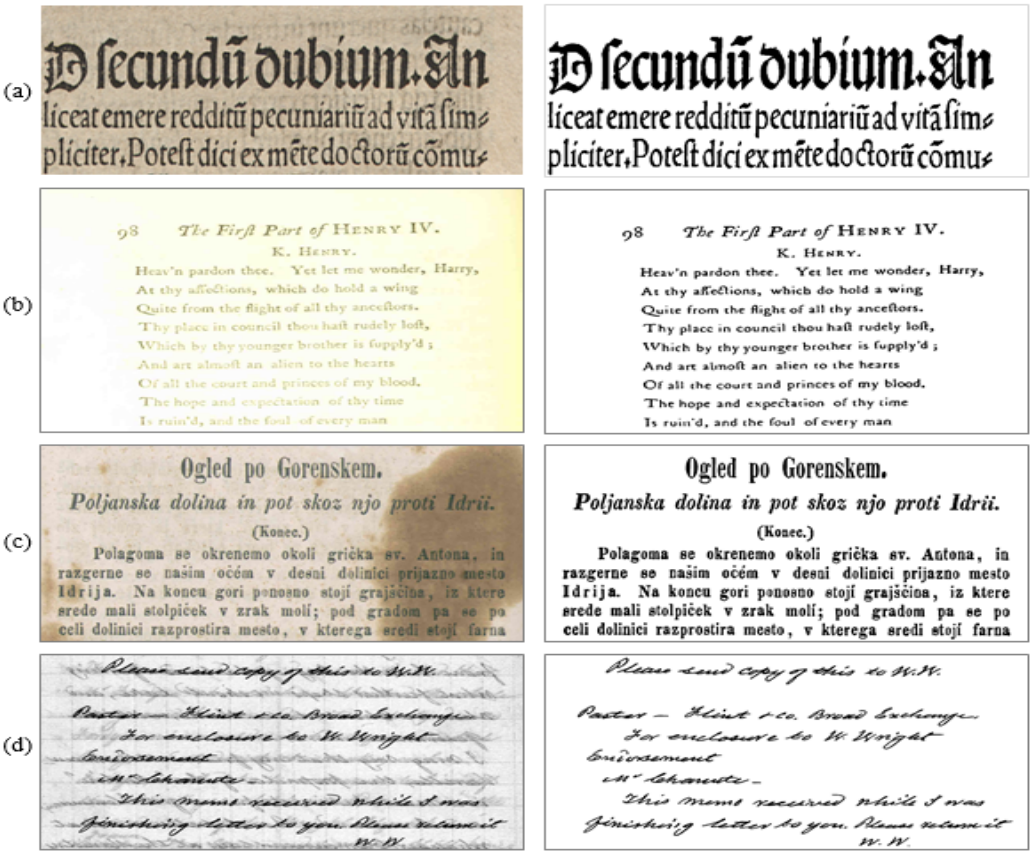


Figure 3.1: Degraded DIBCO images (1^{st} column) with different forms of degradations and their respective binarized forms (2^{nd} column) (ground truth): (a) DIBCO 2009 image (background noise and see throughs) [118], (b) DIBCO 2017 image (uneven illumination)[120], (c) DIBCO 2013 image (artifacts) [119], and (d) DIBCO 2013 image (see throughs) [119]

model by training MLP classifier by backpropagation. Ground truths (binarized forms) of the degraded document images are used as targets during development of the binarization model as is explained in section 3.4.3.

3.4 MLP-based binarization method

The proposed MLP-based binarization technique for HMI has 3 key steps: pre-processing, feature extraction, and training phases as illustrated in figure 3.2. First, historical manuscript images (HMI) from train data set are pre-processed (section 3.4.1) followed by feature extraction where various features are extracted as explained in section 3.4.2. The features are

used to build a binarization model (section 3.4.3). The model is evaluated with test HMIs. The sub-steps are explained in detail in following sections.

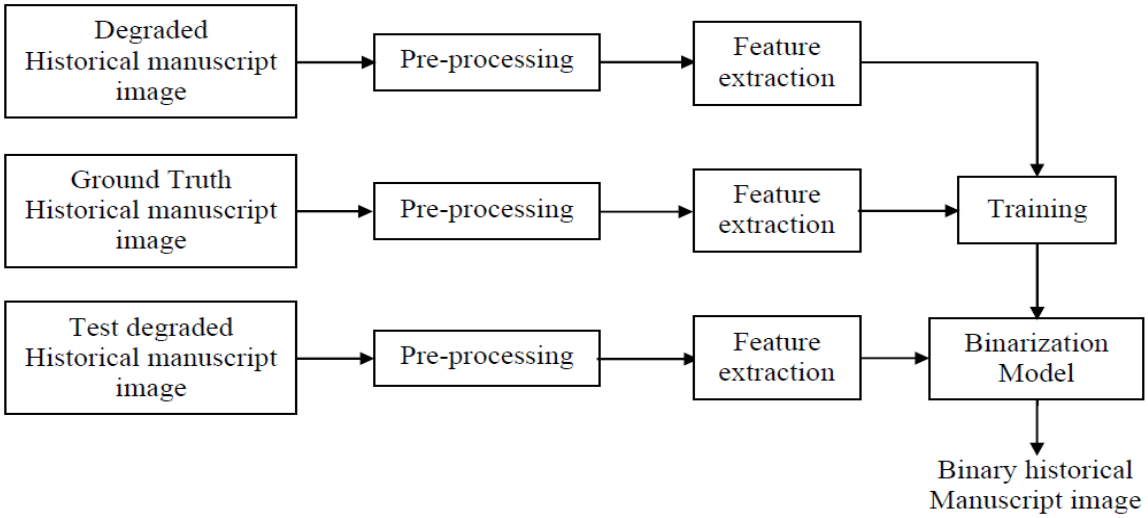


Figure 3.2: Framework of MLP-based binarization for historical manuscript images

3.4.1 Pre-processing

The inputs in this step are degraded 3-level (RGB or color) document images and their ground truths. These images are then converted to gray scale. The degraded gray scale document images are then denoised using mean filters. Filtering is done so that good quality features are extracted in subsequent step.

3.4.2 Feature extraction

Various features are extracted from pre-processed document images (from previous step). These features are per-region fuzzy membership grades of a pixel p_i, local contrast features, local standard deviation, entropy features, and normalized gray levels. Extraction of each feature is explained in detail as follows:

a) Fuzzy membership grade features

Using hard k-means clustering algorithm [121, 122], pixels of a document image are divided into 4 non-overlapping clusters 1-4. The 4 clusters are the fuzzy subregions

of the image. $\mu 1$, $\mu 2$, $\mu 3$, and $\mu 4$ are respectively the means of clusters (fuzzy subregions) 1, 2, 3, and 4 where $\mu_1 < \mu_2 < \mu_3 < \mu_4$. Triangular membership is used to assign membership grades ($\mu(p_i)$) to image pixels (p_i) for various fuzzy regions (clusters) of the image as shown in figure 3.3. From the figure (3.3), the vertical broken lines denote cluster boundaries, $\mu 1$, $\mu 2$, $\mu 3$, and $\mu 4$ are respectively cluster means for clusters 1-4. It should be noted that cluster mean (μ_i) is greater than lower cluster boundary and less than upper cluster boundary. The triangular membership function for a given cluster peaks at a pixel value equal to the cluster mean as shown in figure 3.3. Membership functions for pixels (p_i) for various fuzzy regions are given by equations 3.1-3.3 where μ_j is mean of j^{th} cluster or fuzzy region where $j = \{1, 2, 3, 4\}$. Membership grades (values) ($\mu(p_i)$) for cluster (fuzzy region) 1 is computed using membership function given by equation 3.1. Equation 3.2 is used to compute membership grades for clusters (membership regions) 2 and 3 pixels while equation 3.3 is used to compute membership values for cluster 4 pixels.

$$\mu(p_i) = \begin{cases} 1 & for\ p_i < \mu_j \\ \frac{\mu_{j+1} - p_i}{\mu_{j+1} - \mu_j} & for\ \mu_j \leq p_i \leq \mu_{j+1} \quad for\ j = 1 \\ 0 & for\ p_i > \mu_{j+1} \end{cases} \tag{3.1}$$

$$\mu(p_i) = \begin{cases} 0 & for\ p_i < \mu_{j-1} \\ \frac{p_i - \mu_{j-i}}{\mu_j - \mu_{j-1}} & for\ \mu_{j-1} \leq p_i \leq \mu_j \\ \frac{\mu_{j+1} - p_i}{\mu_{j+1} - \mu_j} & for\ \mu_j < p_i \leq \mu_{j+1} \quad for\ j = \{2, 3\} \\ 0 & for\ p_i > \mu_{j+1} \end{cases} \tag{3.2}$$

$$\mu(p_i) = \begin{cases} 0 & for\ p_i < \mu_{j-1} \\ \frac{p_i - \mu_{j-i}}{\mu_j - \mu_{j-1}} & for\ \mu_{j-1} \leq p_i \leq \mu_j \quad for\ j = 4 \\ 0 & for\ p_i > \mu_j \end{cases} \tag{3.3}$$

Where p_i is i^{th} pixel and $j = \{1, 2, 3, 4\}$ is index of cluster or fuzzy region.

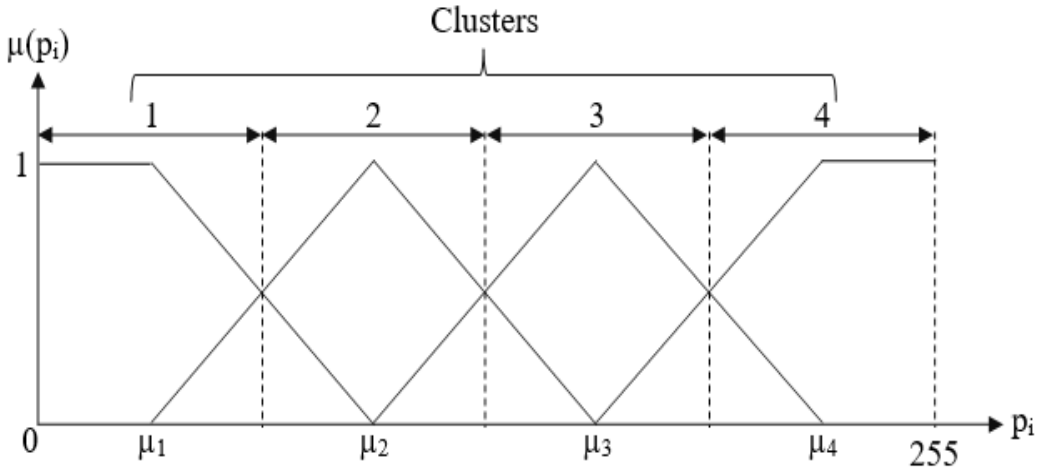


Figure 3.3: Gray values (p_i) fuzzification using triangular membership functions where μ_1, μ_2, μ_3, and μ_4 are respectively means of clusters (fuzzy regions) 1-4 and $\mu(p_i)$ is membership grade for pixel p_i. The broken lines denote cluster boundaries.

For each pixel, 4 fuzzy membership grade values for the 4 clusters are obtained. The membership grades are μp_1, μp_2, μp_3, and μp_4, for clusters 1-4 respectively.

b) Entropy features

For a pixel p_i, entropy feature (ent) is a normalized entropy of pixels of local 5×5 window centered on it. The entropy is given by equation 3.4.

$$ent = -\sum_{i=0}^{n} p_i log_2 p_i \tag{3.4}$$

where n is maximum gray level and p_i is probability of a pixel having gray level i.

c) Contrast feature

Local contrast ($cont(p_i)$) associated with pixel p_i is statistically computed using maximum and minimum gray values in 5×5 window centered in pixel p_i using equation 3.5.

$$cont(p_i) = \frac{p_{max} - p_{min}}{p_{max} + p_{min}} \tag{3.5}$$

The contrast given by equation 3.5 is called Michelson contrast and is also used by Mitianoudis and Papamarkos [18].

d) Local standard deviation feature

This is the feature quantifying the spread of pixels in the neighborhood of a pixel p_i. Standard deviation (σ_l) used is computed by equation 3.6.

$$\sigma_l = \sqrt{\frac{\sum(p_i - \mu_l)^2}{N}} \tag{3.6}$$

Where μ_l is mean of local window and N is total number of local pixels. σ_l is normalized by dividing by largest σ_l.

e) Normalized pixel gray level ($p_n(i)$)

Gray values of pixels are normalized by using equation 3.7.

$$p_n(i) = \frac{p_i}{N} \tag{3.7}$$

Where $p_n(i)$ denotes i^{th} normalized pixel, p_i is gray value of i^{th} pixel that ranges in $0 - 255$, and N is maximum gray value (255). $p_n(i)$ is in $0 - 1$ range.

3.4.3 Model development

A model for binarizing degraded document images has been developed by training MLP with back propagation method. Three layers are used with 1^{st} layer having 16 units, 2^{nd} layer having 8 units, and last layer (output layer) having one unit. The activation function for

layers 1 and 2 is Rectified linear unit (ReLU) whereas sigmoid activation function is used in output layer. Adam (Adaptive moment) optimizer [123] is used for optimization during training phase. The loss function used is binary cross entropy. Batch training is used with batch size of 64. During training, 500 epochs are used.

The MLP-based binarization model has been evaluated with DIBCO (Document Image Binarization Contest) datasets for years 2009, 2011, 2013, and 2017 [118, 119, 120, 124]. Other datasets used are Handwritten DIBCO (H-DIBCO) for 2010, 2012, 2014, 2016, and 2018 [125, 126, 127, 128, 129]. These datasets have been used for binarization contests. The datasets consist of degraded document images and their ground truths. The images have degradations of different forms like fading, see throughs, ink bleeds, poor and uneven illumination, background noise, smudges, artifacts among others. Degraded images are used as input data while ground truths are used as targets. During evaluation, degraded images of one year are held for testing while degraded images of the other years are used for training.

3.5 Evaluation of MLP-based binarization method

In evaluating MLP-based binarization technique, the images used are from DIBCO 2013 [119], DIBCO 2011 [124], and H-DIBCO 2012 [126] datasets. These datasets are used because their degradations are good representation of commonly occurring forms of degradations. The MLP-based binarization method is both qualitatively and objectively evaluated. Objective evaluation is done using 6 ground-truth-based metrics: precision (P) [130], recall (R) [130], F-measure (FM) [130, 131], pseudo F-measure (pFM) [131, 132], PSNR (power to signal noise ratio) [130, 133], and DRD (distance reciprocal distortion) [134]. They are explained in detail as follows:

(i) **Precision (P)**

This is the proportion of relevant pixels that have been correctly identified in test binary image. It is computed using equation 3.8 [130].

$$P = \frac{TP}{FP + TP} \tag{3.8}$$

Where FP is number of false positives and TP is number of true positives.

(ii) **Recall (RC)**

Is the proportion of skeletonized ground truth image (I_G) present in test binarized image. It is given as relevant pixels (object pixels) retrieved divided by relevant pixels in ground truth image (IG) as given in equation 3.9 [130].

$$RC = \frac{TP}{FN + TP} \tag{3.9}$$

Where FN is number of false negatives and TP is number of true positives.

(iii) **F-measure (FM)**

It is computed using equation 3.10 [130, 131].

$$FM = 2 \times \frac{RC \times P}{RC + P} \tag{3.10}$$

Where P and RC are respectively precision and recall metrics. High FM values points to a better the binarization method involved.

(iv) **Pseudo F-measure (pFM)**

It is given by equation 3.11 [131].

$$pFM = 2 \times \frac{pRC \times TP}{pRC + TP} \tag{3.11}$$

Where pRC is pseudo-Recall given by equation 3.12, TP is number of true positives, FP_{skel} and TP_{skel} are respectively numbers of false positives and true positives in skeletonized ground truth image (SG).

$$pRC = \frac{TP_{skel}}{FP_{skel} + TP_{skel}} \tag{3.12}$$

High pFM values points to a better and high quality of binarization method.

(v) **PSNR (Power to signal noise ratio)**

This is the measure of closeness of two images. It is given by equation 3.13 [130].

$$pRC = \frac{TP_{skel}}{FP_{skel} + TP_{skel}} \tag{3.13}$$

Where C is the maximum peak i.e., maximum difference between foreground and background pixels. For uint8 images, $C = 255$, for binarized images with foreground being 1 and background being 0, $C = 1$. Mean square error (MSE) is cumulative squared error between test binarized and ground truth images. MSE is computed using equation 3.14 [131].

$$MSE = \frac{\sum_{i=0}^{M-1} \sum_{j=0}^{N-1} \left(I\left(i,j\right) - I_G\left(i,j\right)\right)^2}{M * N} \tag{3.14}$$

Where I is test binarized image, I_G is ground truth binary image (reference noise-free image), (i, j) is pixel location, M and N are respectively number of rows and number of columns in an image. High values of PSNR scores indicates high similarity of two images I and I_G.

(vi) **DRD (Distance reciprocal distortion)**

This is objective measure of visual distortion between test binarized image and ground truth image (I_G) [134]. For S flipped pixels in test binarized image, it is given by equation 3.15.

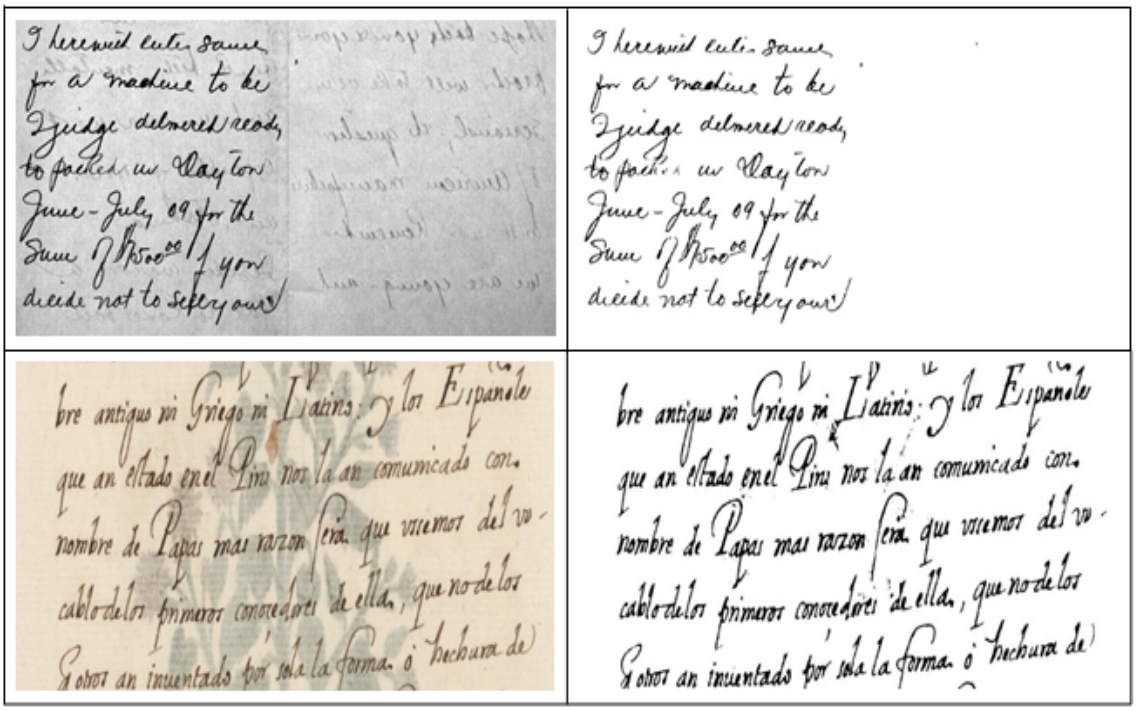
$$DRD = \frac{\sum_{k=1}^{S} DRD_k}{NUBN} \tag{3.15}$$

Where DRD_k is weighted sum of pixels in a block of ground truth image (I_G) that differ from flipped pixel in test binarized image and NUBN is the number of non-uniform (not all foreground/background pixels) 8×8 blocks in (I_G). Small DRD values indicate less distortion between test binarized image and ground truth image hence good quality of binarization process and vice versa. Low DRD indicates high quality of binarization [9].

Evaluation metrics were obtained using H-DIBCO 2016 competition evaluation tool [135].

Figure 3.4 shows degraded images (1^{st} column in Figure 3.4) from DIBCO 2013 [119] and HDIBCO 2014 [127] datasets and their respective binary outputs (2^{nd} column in Figure 3.4) using the proposed technique. The degraded image in 1^{st} row in figure 3.4 has background noise and some see throughs whereas the degraded image in 2^{nd} row in figure 3.4 has artifacts in its background. From the binarization outputs in 2^{nd} column in figure 3.4, it can be seen that background noise, see throughs, and artifacts are successfully removed. This is because the binarization model obtained (in section 3.4.3) is efficient in categorising the image pixels to background and foreground (object) regions. Figure 3.5 shows comparison of binarization outputs of the proposed binarization technique with various state-of-the-art binarization methods/techniques for a degraded image (Figure 3.5a) with background noise. The techniques compared with are those by Otsu [6], Sauvola and Pietikainen [11], Niblack [8], Singh et al. [13], Bernsen [14], Lu et al. [12], Mitianoudis and Papamarkos [18], and Bhowmik et al [136]. From the binary outputs in figure 3.5, it can be seen that there are some bits of background noise in binary outputs of techniques by Otsu [6] (figure 3.5b), Sauvola and Pietikainen [11] (figure 3.5c), Niblack [8] (figure 3.5d), and Singh et al. [13] (figure 3.5e). Some techniques like those by Lu et al. [12] (figure 3.5g), Mitianoudis and Papamarkos [18] (figure 3.5h), and Bhowmik et al [136] (figure 3.5i) have good binary outputs though with very little background noise. Of all these state-of-art methods, the binary output by Lu et al [12] is the best since it has negligible visible background noise. This is because its local method uses local contrast and local mean pixel to obtain local threshold value for binarization which leads to good performance. Bernsen's method [14] (figure 3.5f) gives worst result as there is heavy visible background noise. This is because the approach used for computation of local threshold values causes the background pixels being passed as foreground pixels. The proposed MLP-based method outperforms others as its binary output (figure 3.5j) has no visible background noise. This is because the model obtained models well the background and foreground pixels.

Table 3.1 shows performance of MLP-based binarization technique with DIBCO 2011 [124], DIBCO 2013 [119], and HDIBCO 2012 [126] with precision (P), recall (R), F-measure (FM),



(a) (b)

Figure 3.4: Binarization outputs for proposed method. Column (a) are degraded images from DIBCO 2013 and H-DIBCO 2014 [119, 127] and their binarized outputs in column (b)

distance reciprocal distortion (DRD), pseudo F-Measure (pFM), and power to signal noise ratio (PSNR) metrics. The proposed method achieved average of 96.087%, 93.105%, 94.128%, 94.895%, 20.599, and 3.342 respectively for P, R, FM, pFM, PSNR, and DRD metrics. From the table scores, high values of P, R, FM, pFM, and PSNR and low DRD values show that the proposed method performs desirably well due to modelling well of foreground and background pixels even in the presence of noise.

Table 3.1: Performance of MLP-based binarization method with (H)DIBCO 2011-2013 test datasets [119, 124, 126]

Year	P	RC	FM	pFM	PSNR	DRD
2011	94.826	94.32	94.181	92.937	18.520	3.42
2012	97.055	92.393	93.452	96.207	22.139	3.396
2013	96.381	92.601	94.751	95.541	21.137	3.21
Mean	**96.087**	**93.105**	**94.128**	**94.895**	**20.599**	**3.342**

Performance of MLP-based binarization technique was compared with other state-of-the-art

Figure 3.5: Comparison of binarization results; (a) Degraded image [124], (b) Otsu [6], (c) Sauvola and Pietikainen [11], (d) Niblack [8], (e) Singh et al [13], (f) Bernsen [14], (g) Lu et al [12], (h) Mitianoudis and Papamarkos [18], (i) Bhowmik et al [136], and (j) Proposed MLP-based method.

binarization techniques [6, 11, 14, 15, 18, 136, 137, 138] with the test (H)DIBCO datasets [119, 124, 126]. The comparative performances with mentioned metrics are shown in tables 3.2-3.5 for P, R, FM and PSNR metrics. For DIBCO 2011 [124] (table 3.2), the proposed method has the best performance for P, R, FM, and PSNR metric scores. The method by

Bhowmik et al [136] comes second, followed by Mitianoudis and Papamarkos [18] in same order except for precision metric. Otsu [6] is the least performing method mainly because it is a global method, thus not able to binarize images with uneven illumination, ink bleeds and see throughs.

Table 3.3 shows comparison of performance of the proposed technique with other state-of-the-art binarization techniques for HDIBCO 2012 [126] data set. It can be seen that the proposed technique outperforms all other state-of-the-art methods for P, R, FM, and PSNR metric scores. For precision metric, Gatos et al [15] and Badekas and Papamarkos [138] ranks 2^{nd} and 3^{rd} respectively while Otsu [6] comes last. For recall measure, Mitianoudis and Papamarkos [18] is 2^{nd} whereas Badekas and Papamarkos [138] ranks last. For FM, Bhowmik et al. [136] ranks 2^{nd} followed by Mitianoudis and Papamarkos [18] as Badekas and Papamarkos [138] comes last. For PSNR, Su et al [137] ranks 2^{nd}, followed by Bhowmik et al. [136]).

Table 3.2: Comparison of performance of the proposed binarization method with state-of-the-art methods on DIBCO 2011 printed document images [124]

Method	P	R	FM	PSNR
Otsu [6]	83.55	93.05	87.08	14.84
Bernsen [14]	87.33	84.48	85.56	14.39
Sauvola and Pietikainen [11]	81.31	82.79	81.77	12.9
Gatos et al [15]	88.22	84.00	84.86	15.15
Badekas and Papamarkos [138]	91.00	76.40	82.8	13.61
Su et al [137]	83.57	88.35	81.6	15.55
Mitianoudis and Papamarkos [18]	93.63	88.06	90.68	17.72
Bhowmik et al [136]	93.19	88.84	90.81	17.73
Proposed MLP-based method	**94.83**	**94.32**	**94.18**	**18.52**
Ranking	1st	1st	1st	1st

In comparing binarization performances of the proposed binarization method and other state-of-the-art techniques for handwritten DIBCO 2013 [119], it can be seen that the proposed technique outperforms state-of-the-art methods for P, R, FM, and PSNR metrics as shown in table 3.4. For recall and PSNR metrics, Su et al [137] and Mitianoudis and Papamarkos [18] ranks 2^{nd} and 3^{rd} respectively whereas Sauvola and Pietikainen [11] comes last. Table 3.5 shows how the proposed technique compares with other state-of-the-art binarization

Table 3.3: Comparison of performance of the proposed binarization method with state-of-the-art methods on HDIBCO 2012 handwritten document images [126]

Method	P	R	FM	PSNR
Otsu [6]	77.75	86.47	77.48	15.57
Bernsen [14]	77.89	83.83	76.66	15.61
Sauvola and Pietikainen [11]	85.35	77.77	80.08	16.29
Gatos et al [15]	93.99	74.81	81.78	17.04
Badekas and Papamarkos [138]	93.74	47.66	59.56	14.91
Su et al [137]	93.55	86.92	88.87	19.62
Mitianoudis and Papamarkos [18]	89.22	90.77	89.71	18.72
Bhowmik et al [136]	91.96	89.9	90.99	19.34
Proposed MLP-based method	**97.06**	**91.39**	**94.14**	**22.14**
Ranking	1st	1st	1st	1st

techniques. From table 3.5, it is seen that the proposed technique ranks 1^{st} for P, FM, and PSNR metrics whereas it ranks 2^{nd} for recall metric after that of Su et al [137]. In cases where the proposed binarization technique is outperformed (for recall recall metric), the deviation from the best performing method is very small (2.56%). Badekas and Papamarkos [138] comes last for recall metric. For FM and PSNR metrics, Su et al [137] and Bhowmik et al [136] ranks 2^{nd} and 3^{rd} respectively whereas Badekas and Papamarkos [138] ranks last. For precision metric, Bhowmik et al [136] ranks 2^{nd} followed by Mitianoudis and Papamarkos [18] as Otsu [6] ranks last.

Table 3.4: Comparison of performance of the proposed binarization method with state-of-the-art methods on DIBCO 2013 handwritten document images [119]

Method	P	R	FM	PSNR
Otsu [6]	88.44	77.14	77.69	18.43
Bernsen [14]	85.6	75.02	74.56	17.32
Sauvola and Pietikainen [11]	76.29	80.3	74.13	16.3
Gatos et al [15]	90.12	77.44	79.27	18.77
Badekas and Papamarkos [138]	90.35	64.7	72.81	17.23
Su et al [137]	98.33	87.24	92.07	22.84
Mitianoudis and Papamarkos [18]	94.92	89.48	92.09	21.57
Bhowmik et al [136]	92.08	88.7	90.25	20.79
Proposed MLP-based method	**96.81**	**92.60**	**94.75**	**23.14**
Ranking	1st	1st	1st	1st

Table 3.5: Comparison of performance of the proposed binarization method with state-of-the-art methods on DIBCO 2013 printed document images [119]

Method	P	R	FM	PSNR
Otsu [6]	77.58	92	81.86	14.72
Bernsen [14]	78.75	87.42	80.63	14.42
Sauvola and Pietikainen [11]	83.96	83.53	82.84	14.7
Gatos et al[15]	92.44	84.45	87.78	16.35
Badekas and Papamarkos [138]	91.74	71.07	77.45	14.5
Su et al [137]	91.83	**94.04**	92.43	18.89
Mitianoudis and Papamarkos [18]	92.77	89.79	91.07	17.59
Bhowmik et al [136]	95.01	89.4	92.03	18.37
Proposed MLP-based method	**96.38**	92.60	**94.75**	**21.14**
Ranking	1st	2nd	1st	1st

3.6 Conclusion

A robust and efficient machine learning-based method to binarize degraded document images has been presented. In the presented method, binarization is regarded as a hard classification task where image pixels are put either in background or foreground class (region) using a binarization model. The binarization model is developed by training MLP classifier. Training data consist of features extracted from degraded images and their respective ground truths from (H)DIBCO 2009-2018 data sets. Features used are: fuzzy membership grades, local standard deviation, local contrast, local entropy, and normalized pixel gray levels. DIBCO 2011 [124], HDIBCO 2012 [126], and DIBCO 2013 [119] datasets have been used to evaluate the proposed MLP-based binarization method. The proposed method posted very good binary images as outputs.

Chapter 4

HANDWRITTEN WORD SPOTTING FOR INDEXING HISTORICAL MANUSCRIPTS

4.1 Introduction

This chapter discusses segmentation-based handwritten word spotting (HWS) technique for offline scanned handwritten historical manuscript images (HMI). HWS is carried out using newly devised integral histogram of oriented displacement (IHOD) shape descriptor. The HWS method is applied in indexing HMI. Word spotting is very fundamental in document analysis systems working with historical manuscripts.

4.2 Word spotting

This is a pattern recognition task where similar instances of a given query word text/image are obtained from a large database. Word spotting is one of the many challenging historical manuscript management tasks, like archiving, enhancement, restoration, and storage among others. During word spotting, for a given query word image, other words of similar instances are searched from a database of scanned historical manuscripts.

Word spotting can be done by either manual or automated process. In manual word spotting, a person searches for similar instances of a query word from either actual historical manuscript documents or database of scans of historical manuscripts. The person doing the searching is henceforth referred to as retriever. In this manual approach, the retriever compares query and candidate words (in database or actual manuscripts documents) based on visual characteristics. He/she then selects candidate words he/she feels are of same instance as query word. Manual word spotting is dependent on the retriever's eyesight, judgment, and condition. As a result, the approach is less efficient, prone to errors, not feasible in large scale, time consuming, and wearisome. It may also accelerate degradation when actual manuscripts are used instead of scans. Because of non-objectivity of manual word spotting, automated word spotting approach is a better option.

In automated word spotting approach, computer-based algorithms are used in extracting features from both query and candidate word images. The features of query are then objectively matched with those of candidate word images, and similarity scores computed using some matching techniques. A ranked (in order from most likely to least likely) list of words similar to query word is obtained based on similarity scores. Retrieved candidate word images whose similarity scores are greater than a set threshold value are included in the ranked list of likely similar words [139]. Similarity scores can be obtained using distance measures like cosine similarity, normalization cross correlation [33], Euclidean distance [36, 140], and dynamic time warping [141]. The common features used for word spotting include word shapes in text images, zonal features, and profile-based features among others. Word spotting is accomplished by using features like word shapes in text images, zonal features, and profile-based features among others. A ranked list of similar words is output from the word spotting system. These words have same ASCII equivalent [141]. During word spotting process, a word in a document image is recognized or identified without recognizing individual characters constituting the word. This is unlike OCR (optical character recognition) system, which recognize words by first recognizing individual characters of the word whereas word spotting recognizes the whole word without focusing to individual characters constituting the word.

Word spotting in handwritten documents is a cumbersome task as compared to machine printed documents. This is because in handwritten documents, handwritten words have high cursiveness, similarity, and variability of styles of writing among different writers [142]. The sources of within class variability are: (i) elongation of free stroke endpoints, (ii) hill and dale writing - a case where the general orientation of a handwritten word is at a non-zero angle to horizontal axis of the paper, (iii) unconstrained writing, (iv) variation of writing styles among different writers, and (v) connectedness and unconnectedness of constituent characters in a word for different writers.

Word spotting techniques use handcrafted or learned features. The features are obtained from segmented words/sentences (segmentation-based approach) or directly from whole document image (segmentation-free approach). The features are in turn used to train a suitable classifier for word spotting or used directly by nearest neighbour search method. In developing word spotting models, suitable classifiers used include neural networks [30], multi-layer perceptron (MLP), HMM (hidden markov model) [29], and support vector machine (SVM) [31]. Handcrafted features are extracted using dedicated algorithms whereas learnt features are learnt directly from data using deep learning methods like CNN. Bhaldwaj et al [33] used moment-based features in their segmentation-based word spotting system. Srihari et al [34] used word shape method for word spotting in Arabic and Devanagari scripts. Shekhar and Jawahar [35] used bag of visual words as features and scale invariant feature transform (SIFT) features for word spotting in Bangla scripts. Hassan et al [36] used shape descriptor method for word spotting in Bangla and Hindi document images where precision of 87% was obtained. A convolution neural network (CNN)-based word spotting system has been developed by Sudholt and Fink [30]. Zhang et al [37] used segmentation free technique to carry out word spotting for Bangla handwritten document images. The features that were used to represent key points of word images are called Heat Kernel signature (HKS). Zagoris et al [38] used key points based on local gradient vectors and orientations as features for word spotting. Bhunia et al [40] extracted features from middle zone of text-line images and used them for word spotting. Roy et al [143] used dynamic shape coding descriptor word spotting in Latin and Indic scripts. Word spotting techniques that do not employ models (i.e., learning free based word spotting methods) include techniques by Khursid et al [144]

and Papandreou et al [145].

Segmentation-based handwritten word spotting technique for indexing historical manuscript images (HMI) is presented in this chapter. The proposed system uses integral histogram of oriented displacement (IHOD) shape descriptor extracted from segmented word images. IHOD is used to describe text strokes of scanned document word images at local and global levels. Construction of IHOD shape descriptor is explained in detail in section 4.5.4.2. This chapter also addresses the segmentation challenge (especially in segmenting overlapping and crossing words) using component tracing and association (CTA) technique.

4.3 Problem statement

In segmentation-based word spotting techniques, word segmentation poses great challenge (especially in segmenting overlapping and crossing words) yet it determines the output word spotting performance. In addition, there is the challenge of low performance and high computational cost especially for CNN-based word spotting techniques. A good number of feature descriptors used in word spotting tasks have low discriminating power and low performance. This chapter addresses the segmentation challenge (of overlapping and crossing words) using component tracing and association (CTA) technique. The challenge of low word spotting performance is addressed by developing an MLP-based model using a new IHOD shape descriptor which has high discriminating power and is efficient.

4.4 Word segmentation

In this step, individual words are separated from others in textual document image using component tracing and association (CTA) technique. A word is consists of connected components (CC) of strokes, characters, or a combination of both strokes and characters. Word segmentation for handwritten documents (HWD) often face various challenges like overlapping, document degradation, cursive nature of free-handwriting, irregular inter-line gaps, accents, non-straight baselines, inter-word (between adjacent words) distances, cluttering,

punctuation marks, non-uniform intra-word (between adjacent characters/CC of a word) gaps, and presence of diacritic symbols [146, 147, 148]. In many instances, these factors cause segmentation errors. Of all these challenges, cluttering and overlapping are the most challenging to address. Overlapping occurs when a section of a stroke or a character extends into a region of adjacent word as shown in figure 4.1 (a, b, d, and e).

Overlapping occurs when a section of a character or stroke of a word extends to a region of adjacent word as shown in figure 4.1 (a, b, d, and e). Figure 4.1 shows word images with overlapping where figure 4.1a, shows crossing of strokes of words i and iii. In figure 4.1b, there is presence of crossing in strokes of words i and ii, and touching of strokes of words ii and v. In figure 4.1d, there overlapping without crossing of adjacent words. For figure 4.1e, there is crossing of the word images. Figure 4.1c shows over-segmentation (word i) and under-segmentation (word ii) of crossing words in figure 4.1e which is output of many segmentation techniques in literature. Figure 4f.1f shows full segmentation of crossing words in figure 4.1e achieved by the proposed CTA technique. Cluttering occurs when document text is compact, that is, the lines and or words are near to each other and at times having non-straight baselines. The most challenging words to segment without under segmentation or over segmentation are crossing words (figures 4.1a(i and iii) and 4.1b(i and ii)). This is because at the cross point, stroke portions of adjacent words share pixels, and also extend into regions of other words as shown in figure 4.1(e).

In cases where strokes of a word crosses with strokes of neighbouring word(s), junction branch association (JBA) technique (section 4.4.4) is used to efficiently separated them. JBA technique is grounded on the principle that short/small portions joined at a common (reference) point (J) on a contiguous hand drawn line/stroke are symmetric or almost symmetric about the reference point. This implies that the sections (on either side of reference point J) are mirror replicas (or near mirror replicas) of each other about a mirror line passing through the reference point (J) and is perpendicular to a tangent of either section at the reference point (J). Using this approach, crossing strokes (in crossing words) are easily separated. The method is discussed in detail in section 4.4.4.

The CTA method for word segmentation is composed of 3 main steps: (a) Line segmenta-

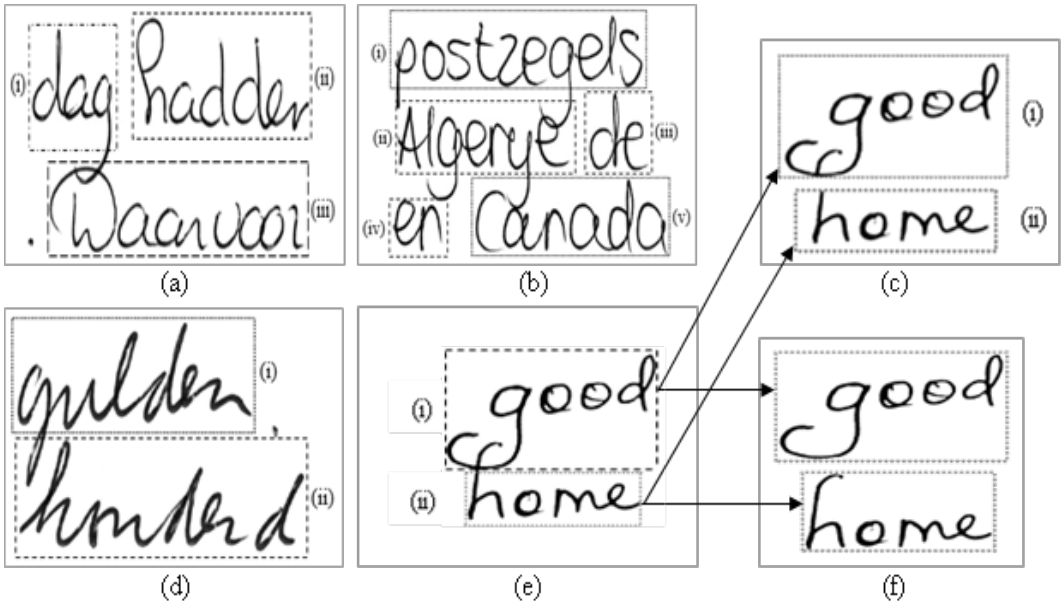


Figure 4.1: Core word segmentation of handwritten words with overlapping and crossings: (a) Crossing words in (i & iii), (b) Crossing words (i & ii) and touching words (ii & v), (c) Over-segmentation (i) and under-segmentation (ii) of crossing words in e, (d) overlapping words, (e) crossing words, and (f) full segmentation of crossing words in e achieved by the proposed CTA technique.

tion, (b) Core word segmentation, and (c) Full word segmentation as shown in figure 4.2. A binarized handwritten image is the input for the proposed CTA method. The line segmentation step is further divided to 4 sub-steps: VSS (vertical stripe stripping), Local horizontal projection profile (HPP) computation, PPS (projection profile smoothing), and LSJ (Line separator joining). Full word segmentation is a refinement of core word segmentation step. The steps are explained in the following sub-sections.

4.4.1 Line segmentation

Text document image is segmented to text lines in this step. It is assumed that the input document image (I_{bin}) has been pre-processed and is binarized with foreground and background pixels respectively represented as 1 and 0. This step consists of 3 sub-steps as already mentioned before: (i) Local horizontal HPP computation, (ii) PPS, and (iii) LSJ.

Local horizontal projection profile (HPP). In this sub-step, I_{bin} is first split into vertical stripes $k = \{1, 2, ... |k|\}$, in a process called vertical stripe splitting (VSS). Figure

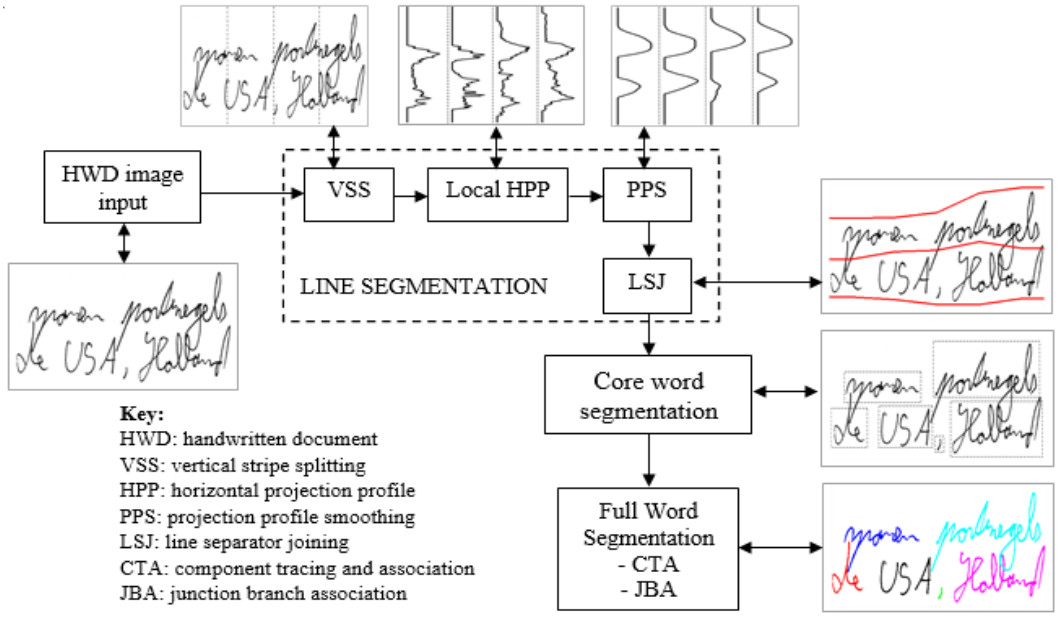


Figure 4.2: Framework for CTA technique for word segmentation where the input is a binarized handwritten word image.

4.3a(i) shows a sample vertical stripe of a handwritten document (HWD). It is assumed that text lines of a vertical stripe are straight or nearly straight (Figure 4.3a(i)), hence making projection profile process feasible. Raw local horizontal projection profile (HPP) of 1^{st} strip (i.e., $k = 1$) is then computed. Figure 4.3a(ii) shows the plot of a raw (full/black) local HPP.

Projection Profile smoothing (PPS). This is the process of refining local HPP of vertical stripes so as to bring out well the valleys between text lines as shown in figure 4.3a(ii) for blue/dashed plot which is a smoothed version of raw local HPP (full/black plot in figure 4.3a(ii)). PPS process is borrowed from that of Papavassiliou et al [149] with some improvements to it for better results. Weights are applied to summation of N profiles in the neighbourhood of a given raw profile (P_i) to obtain PPS. In PPS process, only rows with object/text pixels greater a set threshold (P_{th}) (main rows) participate while those with text pixels less than P_{th} (marginal rows) are ignored. Marginal rows are rows which are empty (or have very few text pixels) and are mostly in inter-text line spaces. P_{th} is computed from HPP of vertical stripes as a median profile in a list of profiles of all rows.

Smoothed profile (SP_i) of i^{th} projection profile (P_i) is computed using equation 4.1 (Papavassiliou et al [149].

$$SP_i = \sum_{j=-N}^{N} d_{i+j} w_j P_{i+j} \tag{4.1}$$

Where N is the number of steps/profiles in the neighbourhood of current projection profile (P_i), $j = \{-N, -N+1, ..., N\}$, and d determines if a given row takes part or is ignored as given by equation 4.2, and w_j are weights computed using equation 4.3.

$$d = \begin{cases} 1 & if\ P \geq P_{th} \\ 0 & Otherwise \end{cases} \tag{4.2}$$

$$w_j \ = \ exp\left(-\frac{|j|^2}{2\,|j|+1}\right) \tag{4.3}$$

It should be noted that w_j has a better and smoother exponential decay with distance away from a given projection profile P_i as compared to the one used by Papavassiliou et al [149]. Therefore, w_j ensures that contribution of elements to SP_i decreases with distance from current profile P_i. SP_i helps to mitigate the problem of empty rows (with no text pixels) or rows with insufficient text pixels. With w_j decaying exponentially away from current profile (P_i) on either side and marginal rows disregarded, text lines and inter-line valleys are well brought out, hence easily recognized as can be seen from figure 4.3a(ii) for dashed/blue plot. Final PPS is obtained by computing integral of the smoothed profiles (SP_i) using equation 4.4.

$$\Delta SP_i(j) = \frac{1}{2h+1} \sum_{x=1}^{h} (\ SP_i\,(j+x)\ -\ SP_i\,(j-x)) \tag{4.4}$$

Where $i = 0, 1, 2, ..., H$ is index of smoothed projection profile with H being height (number of rows) of input document image or stripe, and h is near odd integer to half of mean height of all connected components (CCs) in a HWD (handwritten document). All values of $\Delta SP_i(j) < 0$ are replaced with 0 so as to bring out well text boundaries and their attributes. The integral $(\Delta SP_i(j))$ can further be bettered by re-smoothing them using equations 4.1-4.3 to enable text spatial attributes come out very well. This also help to mitigate the problem

of false local extremas present in a related method proposed and used by Papavassiliou et al [149]. It can be seen from figure 4.3a(iii) that a plot of $\Delta SP_i(j)$ depicts 3 main local turn points that make a recurring sequence, i.e., L_u (up-turn point), L_p (local maxima point), and L_d (down-turn point). L_u is a local point where $\Delta SP_i(j)$ changes from 0 to larger values, L_p is local maxima, and L_d a local point where $\Delta SP_i(j)$ changes from larger values to zero. L_d is a point where principal text line passes. A principal line here refers to line that passes through the center of middle zone of a text/word or CC. L_p is a point where headline (i.e., upper row of central zone) passes. It should be noted that L_u points are valley points (empty spaces or spaces between lines having very few foreground pixels). Text line boundary points are consecutive L_u points as already described. Therefore, the L_u points make text line separators for k^{th} vertical stripe as shown in figures 4.3a(iii-iv). Line separators for all vertical stripes are obtained using the same process.

Line separator joining (LSJ). This is the process of joining corresponding L_u points (line separators) of adjacent vertical stripes to obtain line segmentations for entire handwritten document image as shown in figure 4.3a(iv). Figure 4.3(b) shows a full HWD with line separators.

4.4.2 Core word segmentation

It is assumed that word segmentation at this point consists of core/essential portions of target word, mainly from middle zone of the word. First, CCs in every line segment are obtained. On the basis of gaps between adjacent CCs, words are obtained by clustering the adjacent CCs.

Inter-CC gaps are modelled as tri-variate Gaussian mixture model (GMM) using expectation maximization technique [150]. As a result, a gap is determined to be either intra or inter-word gap. Therefore, the data used has 2 Gaussians: one for inter-word gaps and another for intra-word gaps. The data consists of 3 distances (for gaps) between pairs of adjacent CCs in a line segment: hull distance (d_h), principal hull distance (d_{ph}), and bound box distance (d_b). An inter-CC gap g_i is represented and described by 3 variables (gap distances) $g_i = \{d_h, d_b, d_{ph}\}$. Bound box distance ($d_b$) is the distance between bounding boxes of 2

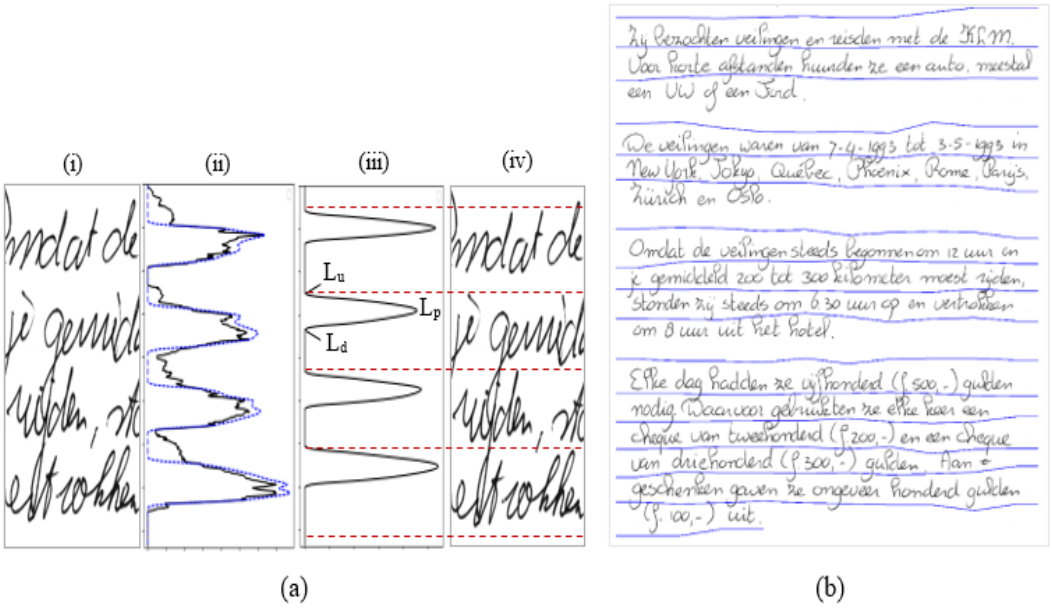


Figure 4.3: (a)Line segmentation of a vertical stripe of handwritten document (HWD) where: a(i) is vertical stripe of HWD, a(ii) raw (full/black line) and smoothed (dashed/blue line) horizontal projection profiles of vertical stripe in a(i), a(iii) is integral of smoothed profile in a(ii) showing up-turn point (L_u), local maxima point (L_p), and down-turn point (L_d), and a(iv) is a vertical stripe with line segmentations. (b) is handwritten document showing line boundaries.

adjacent CCs as shown in figure 4.4. Hull distance (d_h) between 2 adjacent CCs is computed as follows: first obtain convex hulls of the 2 adjacent CCs. Then find the centres (C_1 and C_2) of gravity of the 2 convex hulls. Join the 2 centres of gravity (C_1 and C_2) with a straight line. The distance between points where line C_1C_2 intersects convex hulls of the 2 CCs is the hull distance (d_h) [151, 148]. The inter-CC gap distances are good evaluation metrics for intra and inter-word/CC gaps. Distance d_{ph} is a newly proposed gap metric that factors in non-uniform base lines of handwritten scripts. It is computed as follows: For 2 horizontally adjacent CCs/words, obtain convex hull of each as shown in figure 4.4. For each CC/word, obtain 2 intersection points where its principal line intersects with its convex hull. Principal line is a horizontal line passing through the middle of central zone of a word/CC as shown in figure 4.4. The Euclidean distance between the right intersection point for CC/word on the left and left intersection point for the CC/word on the right is the principal hull distance (d_{ph}) as shown in figure 4.4.

To model the inter-connected component gaps as tri-variate Gaussian mixture model

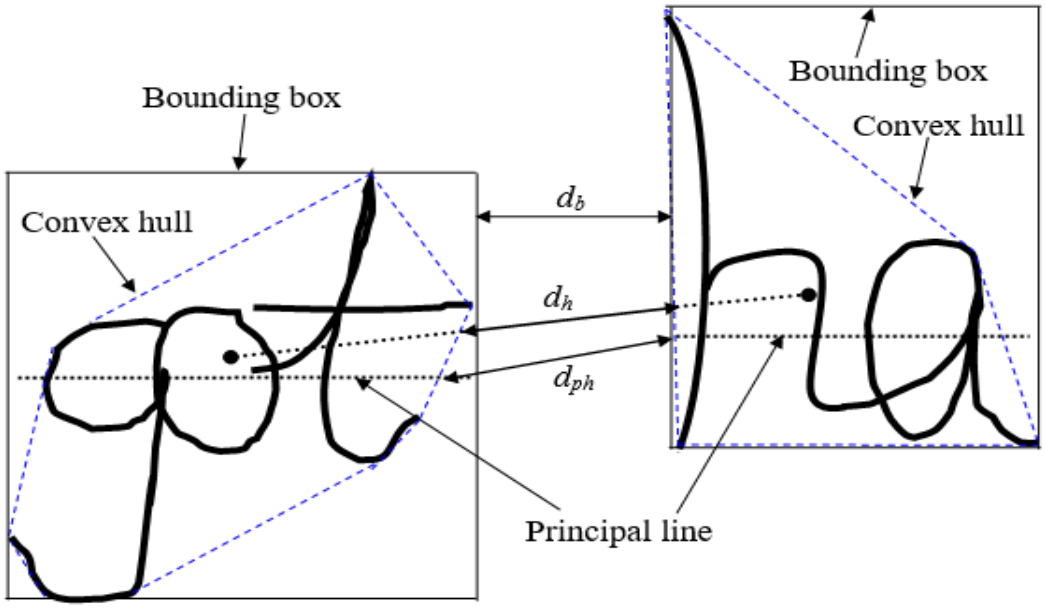


Figure 4.4: Handwritten adjacent connected components "got" and "ha" showing convex hulls (dashed/blue lines), bounding box (full/black lines), principal lines, hull distance (d_h), principal hull distance(d_{ph}), and bound box distance (d_b).

(GMM), probability density function used is shown in equation 4.5.

$$P\left(g_i|\mu,\Sigma\right) = \frac{1}{\left(2\pi\right)^{D/2}\left|\Sigma\right|^{1/2}} exp\left(-\frac{1}{2}\left(g_i-\mu\right)^T\Sigma^{-1}\left(g_i-\mu\right)\right) \tag{4.5}$$

Where $i = 0, 1, 2, ..., M$ is gap index of inter-CC gaps in a given text line with M being number of CCs in the given text line, $g_i = \{d_h, d_b, d_{ph}\}$ is vector of metrics/distances of i^{th} gap between adjacent CCs in a given text line, $\mu = \{\mu_h, \mu_b, \mu_{ph}\}$ is vector of means of each of gap metric from entire text document with μ_h, μ_b, μ_{ph} being means of d_h, d_b, d_{ph} metrics respectively, $\sum$ is a $D \times D$ covariance matrix for gap distances (g_i) found in the whole text document, $D = 3$ is dimension of data (equivalent to number of inter-CC gap distances between 2 adjacent CCs), $\pi = 3.142$ is a mathematical constant, and T symbolises transpose.

After modelling of inter-CC gaps is complete, parameters obtained are mixing coefficients $\Pi = \{\Pi_1, \Pi_2\}$, cluster means $\mu = \{\mu_1, \mu_2\}$, and cluster covariances $\Sigma = \{\Sigma_1, \Sigma_2\}$ which are associated with each cluster/Gaussian. Using these parameters, assignment scores (r_k) are computed for every gap g_i using equation 4.6. These assignment scores are posterior

probabilities for gap g_i belonging to both Gaussians/clusters.

$$r_k\left(g_i\right) = \frac{P\left(k\right) P\left(g_i|k\right)}{P\left(g_i\right)} = \frac{\Pi_k P\left(g_i|\mu_k, \Sigma_k\right)}{\sum_{j=1}^{K} \Pi_k P\left(g_i|\mu_j, \Sigma_j\right)} \tag{4.6}$$

If a gap g_i gets the largest assignment score (posterior probability) for a given cluster/Gaussian, it is assigned to it. Thus, a given gap g_i is classified as inter-word or intra-word gap. A pair of adjacent CCs are grouped together if the gap between them is classified as intra-word. A pair of adjacent CCs belong to different words if the gap between them is classified as inter-word gap. The segmented word obtained at this point is called core word segment (CWS) because it consists of core portions of the word which in most cases is central portion/zone of the word. The CWS may or may not be a full target word. It can either be partial CWS (PCWS) or full CWS (FCWS). PCWS is a CWS whose parts of target word are left-out as shown in figures 4.1a(i), 4.1b(ii), and 4.1d(i). PCWS occur due to range of factors like overlapping, touching, and crossing of words. FCWS is a CWS consisting only of all parts of a target word like in figures 4.1a(ii) and 4.1b(iii). It should be noted that FCWS has no parts from other words.

PCWS are processed further so as to obtain a full target word using component tracing and association (CTA) technique as is discussed in detail in section 4.4.5. CTA employs $MD - DTW_D$ (multi-dimensional dynamic time warping with dependence) (section 4.4.3) and junction branch association (JBA) (section 4.4.4) techniques to handle crossings and junctions that are a common occurance in freely handwritten words. Thus, JBA and $MD - DTW_D$ techniques are discussed first.

4.4.3 Multi-dimensional dynamic time warping with dependence ($MD - DTW_D$)

$MD - DTW_D$ is a technique used to compare multi-dimensional sequences which need not be equal in length. Sequence length refers to number of elements in any one of its dimensions. All dimensions of the same sequence must be of same length. With $MD - DTW_D$, it is assumed that dimensions in a give sequence depend on one another [152]. Multi-dimensional

sequences A and B are represented by equations 4.7 and 4.8.

$$A = \{a_{i,k}\} \tag{4.7}$$

$$B = \{b_{j,k}\} \tag{4.8}$$

Where $a_{i,k}$ denotes i^{th} datapoint/element in k^{th} dimension in multi-dimensional sequence A, $i = 1, 2, ..., m$ is index of an element in sequence A, m is length of sequence A, $b_{j,k}$ denotes j^{th} datapoint/element in k^{th} dimension in multi-dimensional sequence B, $j = 1, 2, ..., n$ is index of a datapoint in sequence B, n is length of sequence B, $k = 1, 2, ..., L$ is index of dimension of sequence A/B, and L is number of dimensions in sequence A/B. It should be noted that both sequences must have equal number of dimensions.

In $MD - DTW_D$ process, a table of L1 distances $D(i, j)$ between elements in sequences A and B is obtained. Equation 4.9 is used to compute $D(i, j)$ [152].

$$D\left(i,j\right) \;=\; d\left(a_i, b_j\right) \; + min\left[D\left(i-1, j-1\right), D\left(i, j-1\right), D\left(i-1, j\right)\right] \tag{4.9}$$

Where $j = 1, 2, ..., n$ is index of a datapoint in sequence B, $i = 1, 2, ..., m$ is index of a datapoint or element in sequence A, $d\left(a_i, b_j\right) = \sum_{k=1}^{L} |b_{j,k} - a_{i,k}|$ is summation of L1 distances between a pair of datapoints a_i and b_j respectively of sequences A and B for all dimensions, and $k = 1, 2, ..., L$ is index of dimension where L is the number of dimensions in sequence A/B. From the table of $D(i, j)$, a table of accumulated cost ($Cost(i, j)$) is obtained using equation 4.10 [152]. Then the warping cost (Wcost) between sequences B and A is obtained using equation 4.11.

$$Cost\left(i,j\right) \; = Cost\left(i,j\right) \; + min\left[Cost\left(i, j-1\right), Cost\left(i-1, j\right), Cost\left(i-1, j-1\right)\right] \tag{4.10}$$

$$Wcost = \frac{Cost(m,n)}{m*n} \tag{4.11}$$

Where m, n, i, and j assume initial meanings. The minimum value of $Wcost$ is 0. $Wcost$ is the similarity score of the sequences compared. Low values of similarity score ($Wcost$) indicates that the two sequences compareed are similar whereas large similarity score ($Wcost$) means the two sequences are not dissimilar. When $Wcost$ is 0 it means the two sequences are perfectly same.

4.4.4 Junction branch association (JBA) method

Junction branch association (JBA) is a technique for segmenting crossing words by identifying pair of junction branches at a cross point belonging to a word. JBA method is applied at crossing points (junctions) where strokes of adjacent words cross/intersect as shown in figure 4.5(a) for strokes of words *"good"* (i) and *"home"* (ii). Figure 4.5(b) shows enlarged cross point/intersection shown in figure 4.5(a). Cross-point region should cover around 5-10 pixels from cross point along junction branches.

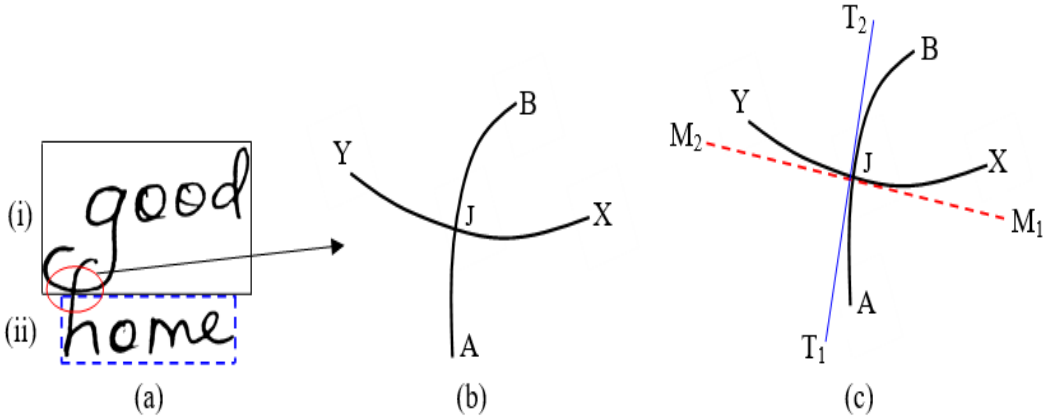


Figure 4.5: (a) A pair of words with crossing strokes at a junction point circled, (b) junction point (J) shown for crossing words in (a) consisting of 4 junction branches: XJ, JY, AJ, and JB, (c) Crossing strokes showing a tangent (T_1T_2) to AJ at J and mirror line (M_1M_2) for AJ

From figure 4.5(b & c), letting (x_j, y_j) be coordinates of cross point J and (x_i, y_i) be coordinates of various points on junction branches (candidate branches), following steps are followed:

(i) Using AJ (in figure 4.5(b)) as a search/object branch, obtain a tangent (T_1T_2) of AJ at cross point J as shown in figure 4.5(c). Compute gradient (g_t) of the tangent T_1T_2.

(ii) Make a line perpendicular to the tangent T_1T_2 obtained in step (i) and passing through cross point $J(x_j, y_j)$. This is mirror line ml for search branch AJ (i.e., M_1M_2 in figure 4.5c). Compute gradient (g_m) of the mirror line M_1M_2. Also compute y (row) intercept (c_m) for mirror line (M_1M_2) using equation 4.12.

$$c_m = y_j - g_m x_j \tag{4.12}$$

Where x_j and y_j are respectively x and y coordinates of cross point J and g_m is gradient of mirror line (M_1M_2).

(iii) Flip the search branch AJ about mirror line M_1M_2 to obtain mirror image AJ' as follows: For an object point (x_i, y_i) on search branch AJ, its mirror point (x_m, y_m) about mirror line M_1M_2 is obtained using equations 4.13 and 4.14.

$$x_m = \frac{2y_i \; - \; x_i \; (g_m \; - \; g_t) - 2c_m}{g_m \; - \; g_t \; + \; \varepsilon} \tag{4.13}$$

$$y_m = y_i \; - \; g_t \; (x_m \; - \; x_i) \tag{4.14}$$

Where (x_i, y_i) are coordinates of an object point on search branch AJ, g_m is gradient of mirror line (M_1M_2) for AJ, g_t is gradient of tangent of AJ at cross point (J), ϵ is regularization value (very small positive value) that prevents division by zero, and c_m is y(row) intercept for mirror line (M_1M_2). Repeat the process for all other object points on AJ and obtain their respective mirror points of mirror branch AJ'.

(iv) Sequence of coordinates of points on AJ'(mirror of AJ about M_1M_2) is compared with each of the sequences of coordinates of points on JB, XJ, and JY junction branches using $MD-DTW_D$ [152] (section 4.4.3). Sequence of coordinates of points are such that they start from cross point (J) outwards. The warping cost ($Wcost$) of $MD-DTW_D$

technique [152] between AJ' and each of the other (candidate) junction branches is computed using equations 4.9-4.11. This *Wcost* is the associativity score. A candidate junction branch in which lowest associativity score is obtained and is less than a set threshold (1) is deemed to be an associate of AJ (search branch). Small associativity scores mean the two sequences (of AJ' and a candidate branch) are similar. Large associativity scores mean the two branches are dissimilar. The minimum associativity score is 0. If no associate branch is found, it means the stroke terminates at the cross point.

Associate of any other junction branch is obtained in same way.

4.4.5 Full word segmentation

Full words are segmented without under/over-segmentation by applying component tracing and association (CTA) technique. The inputs in the CTA techniques are CWS_i (i_{th} core word segment) of a given text line segment where $i = 1, 2, ..., N$ is index of CWS in a segment text line. CTA is composed of 3 main phases: CWS_i classification, connected component tracing (CCT), and JBA as illustrated in CTA framework shown in figure 4.6. In CTA framework (figure 4.6) for word segmentation the input is code word segment (CWS) from segment lines. First, the CSW_i is classified to either partial CWS (PCSW) or full CSW (FCWS) by CWS classifier (figure 4.6). FCWS are regarded as full segmented words. The strokes of PCWS that are left out are traced in a process called connected component (CC) tracing and joined to parent PCWS. During tracing, if a crossing/junction is encountered, JBA (section 4.4.4) is performed to segment the crossing parts. The traced path is de-skeletonized and then joined to parent PCWS to obtain full segmented word. The steps are explained in detail in paragraphs that follow.

For full word segmentation by CTA approach, the following steps are followed:

(i) HWD image is thinned to get a thinned binary image (I_{th}).

(ii) CWS_i is categorized to either full (hence FCWS) or partial (hence PCWS) by CWS-classifier as follows: obtain connected components (CC_r) in a small region (like 20

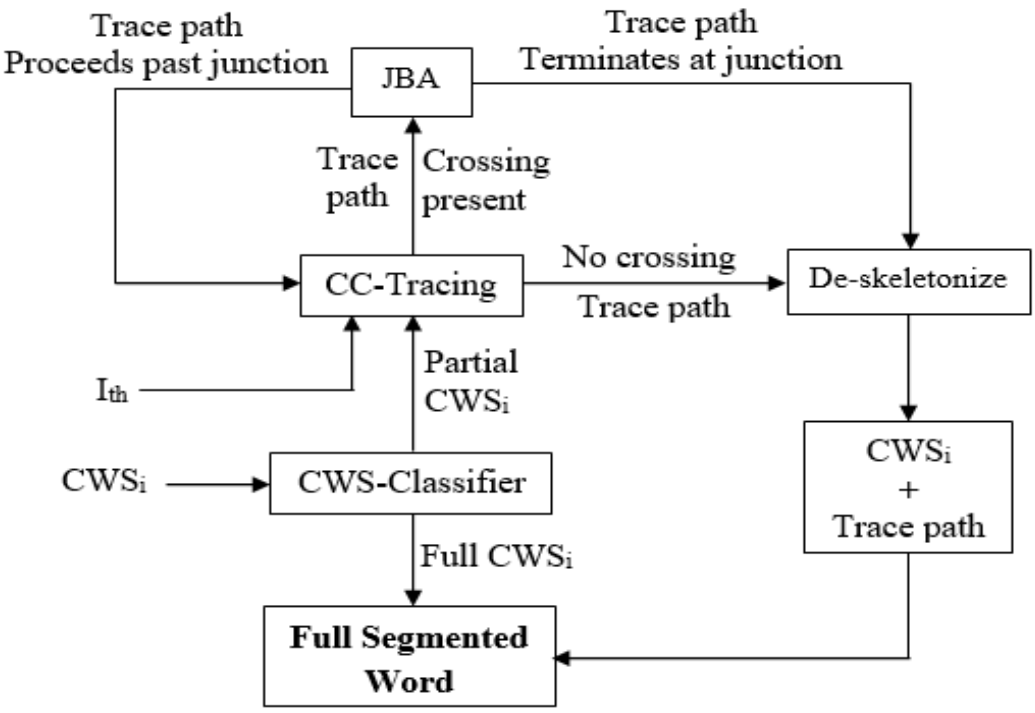


Figure 4.6: Framework of CTA technique for full word segmentation without under/over-segmentation. The inputs are CSW_i and thinned HWD image (I_{th}) while the output is full segmented word

pixels) around region occupied by CWS_i in I_{th}. Let the CC_r (if present) be referred to as neighbouring CC (CC_N). Presence of CC_N connected to CWS_i by 8N connectedness indicates that CWS_i is PCWS. Otherwise, CWS_i is FCWS.

(iii) If CWS_i is categorised as FCWS, it is deemed to be a full segmented word. If CWS_i is not the last one in its line, get the next CWS (i.e., CWS_{i+1}) and go back to step (ii). If CWS_i is categorised as PCWS, it means some of its sections are left out. Therefore, continue with the next step (iv) so as to search for the portions left out.

(iv) The foreground pixels connected to CWS_i (in I_{th}) are traced out using 8N connectedness approach, a process called connected component tracing (CCT). During CCT process, if the trace-path terminates without encountering a junction/crossing, the trace-path that consists of traced-out pixels are de-skeletonized and then joined to the parent CWS_i so as to obtain a full segmented word. If CWS_i is not the last one in its line (i.e., $i < N$), get the next CWS (i.e., CWS_{i+1}) and go back to step (ii). If a

crossing or junction is met, proceed to next step (v).

(v) Apply JBA (junction branch association) technique (section 4.4.4) to junction branches to identify junction branch belonging to CWS_i and also determine if the trace-path (belonging to CWS_i) terminates at the junction. If the trace path ends at the junction, de-skeletonize and join it to CWS_i so as to obtain a full segmented word. If CWS_i is not the last one in its line (i.e., $i < N$), get the next CWS (i.e., CWS_{i+1}) and go back to step (ii). If trace path extends beyond cross point, a junction branch (called associate branch) that is a continuation of trace path up to the junction is identified by JBA technique (section 4.4.4) and then joined to trace path. The other branches are ignored as they don't belong CWS_i. Using the last pixel in the trace path so far as the start point, go back to step (iv).

Repeat steps (ii) – (v) for all CWS from all line segments so as to obtain full word segments from the entire HWD.

4.5 Handwritten word spotting (HWS)

The proposed handwritten word spotting (HWS) method has four main steps: (a) pre-processing, (b) word segmentation, (c) IHOD feature extraction, and (d) training as shown in the Handwritten word spotting (HWS) framework shown in Figure 4.7. These steps are explained in detail in sub-sections that follow. Figure 4.7 shows framework of the proposed HWS technique. The system takes input query word image, pre-processes the word image, extracts IHOD features, and then trains a multilayer perceptron (MLP) classifier. After training, MLP-based model is obtained, which is then used to find words of similar instances from a given database. During testing, features same as the ones used during training phase are extracted from segmented test word images, and then fed to word spotting model to obtain words of same instance from the database.

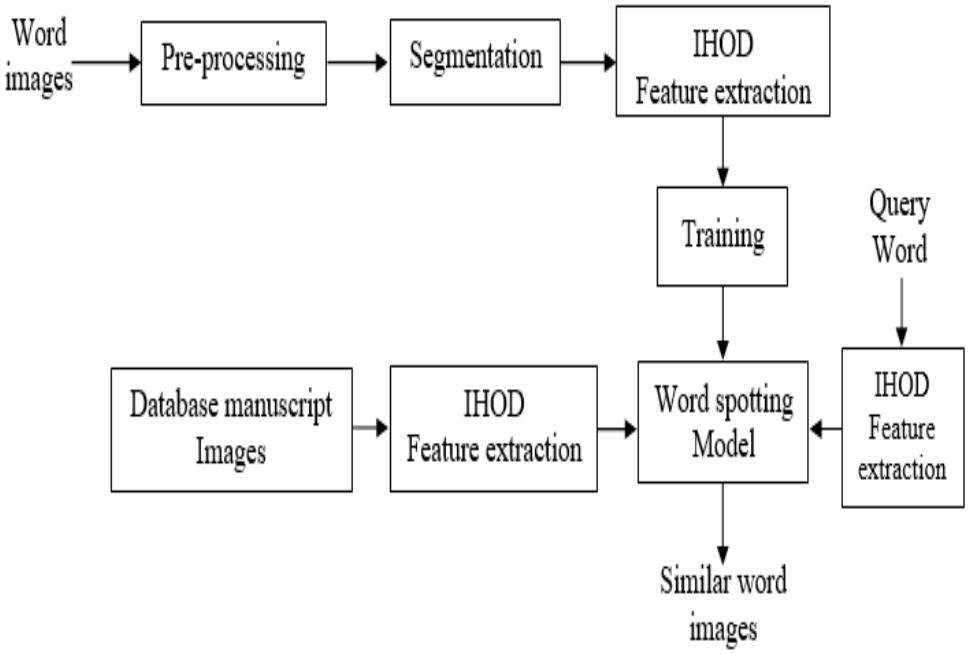


Figure 4.7: Handwritten word spotting (HWS) framework

4.5.1 Pre-processing

Most document images suffer degradations of various forms like poor contrast and noise. Pre-processing is needed for proper feature extraction. In the proposed technique, word images are the inputs to HWS system. For colour image inputs, color-to-gray conversion is carried out to obtain a gray scale image. Mean filter masks are used for noise filtering. The query word image is then binarized by Otsu method ([6]) such that background (non-text region) is 0 and foreground (text region) is 1.

4.5.2 Word image segmentation

For HWS system, document images are segmented to obtain individual words. Segmentation is done using CTA method described in section 4.4.5.

4.5.3 Normalization

Freely handwritten words suffer from skew and slantness. Skew and slantness of segmented word images are detected and corrected accordingly to remove variation. Skew refers to the angle formed by horizontal axis of a word image and horizontal axis of the document image. Skew correction is performed to align text orientation with its axes. Projection profile is used as explained in the procedure below.

(i) Rotare the word image about its center in the range -50^o to 50^o in angle steps of 1^o.

(ii) In each angle step, obtain horizontal projection profile.

(iii) The angle (θ_{sk}) for which horizontal projection profile has highest peak is the skew of the word image.

(iv) Perform skew correction by rotating the word image by angle equivalent to skew angle but in opposite direction, that is $-\theta_{sk}$.

Slantness is the orientation of vertical strokes of a word with respect to horizontal axis of the word. Projection profile is used for slant correction as explained below.

(i) A word image is rotated about its center in the range -50^o to 50^o in angle steps of 1^o.

(ii) In each angle step, obtain vertical projection profile.

(iii) The angle (θ_{ds}) for which vertical projection profile has highest peak is the de-slant angle. Slanting angle (θ_s) is obtained by subtracting θ_{ds} from 90.

(iv) Perform slant correction using local centered-slant correction approach. θ_{ds} is used as global de-slant correction angle.

Local centered-slant correction: A foreground (text) pixel is rotated at an angle $-\theta_{ds}$ about a local center (figure 4.8). Local center is the center of word's bounding box, that is a point where middle horizontal and middle vertical axes of a word's bounding box intersect as shown in figure 4.8. Thus, local center is on middle of a word's horizontal axis. The

acute angle formed by the line connecting foreground pixel and local center, and middle axis is slant angle (θ_s) as shown in figure 4.8. This slant correction ensures preservation of stroke size of word image. After normalization, morphological dilation and erosion are performed to fill any resulting gaps. Figure 4.9 shows normalized word images obtained by local centered-slant correction method for the skewed word images in 1^{st} column. It can be seen that desirable normalization is obtained.

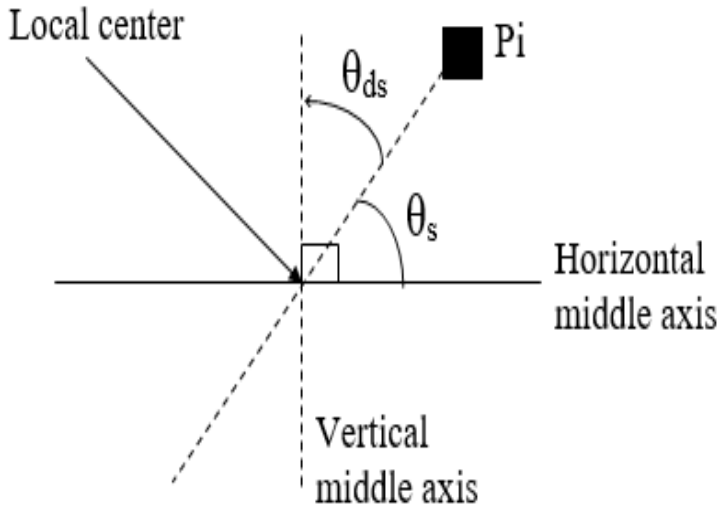


Figure 4.8: Local centered-slant correction for a foreground pixel Pi where θ_s is slant angle and θ_{ds} is de-slanting angle.

4.5.4 IHOD feature extraction

In this section, a 1D shape descriptor called integral histogram of oriented displacement (IHOD) is introduced. IHOD is a combination of 2 histograms of oriented displacement that describe text strokes at both local and global levels as will be explained. Displacement is computed as a distance between foreground pixels and local (cell) or global centres. For local displacements, the word image is subdivided into cells of fixed width and height dimensions. For global displacements, the word image is not sub-divided into rectangular cells. Displacement of foreground pixels (in each cell or entire image) from their respective centres are described by histogram of directions between them (foreground pixels) and the centres. The first step in the construction of IHOD descriptor is standardization of word image.

Figure 4.9: Normalization of word images by local centered-slant correction method. Left column shows skewed word images and right column are their respective normalized and binarized word images

4.5.4.1 Word image standardization

For word spotting task, word images ($I(x, y)$) are resized to standard 45×135 image ($I_f(x, y)$) that is 45 rows (height) by 135 columns (width). Standardization of word image is obtained with pseudo equation 4.15. This will ensure uniform size of extracted features.

$$I_f\left(x, y\right) = Resize\left(I\left(x, y\right), dim\right) \tag{4.15}$$

Where $dim = (135, 45)$ is resizing dimension.

4.5.4.2 IHOD computation

There are 5 steps in construction of IHOD shape descriptor: (a) computation of cell histogram of oriented displacement (CHOD), (b) local histogram of oriented displacement (LHOD), (c) block histogram of oriented displacement (BHOD), (d) global histogram of oriented displacement (GHOD), and (e) computation of IHOD by concatenation of LHOD

and GHOD as shown in figure 4.10. In the figure (figure 4.10), the upper section obtains local features whereas the lower section obtains global features. The steps are explained in the following sub-sections.

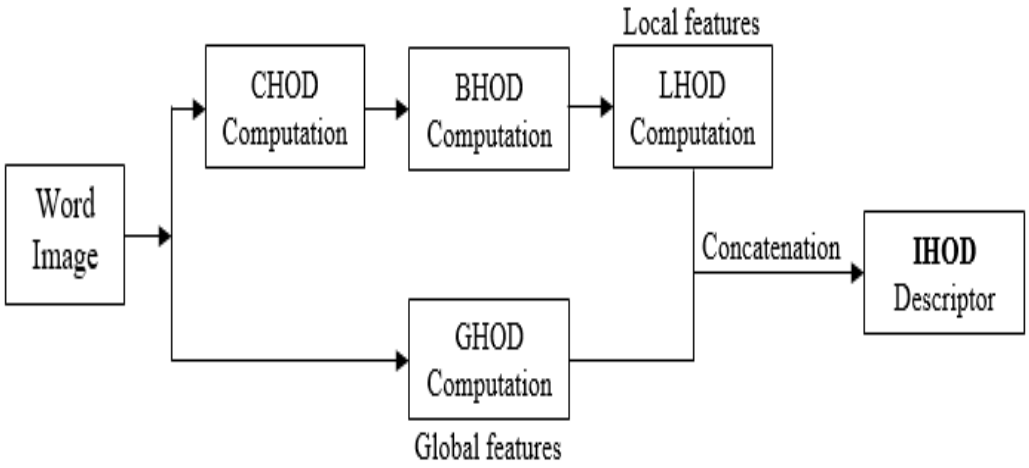


Figure 4.10: Framework of IHOD shape descriptor

(a) **Computing cell of oriented displacement (CHOD)**

In computing CHOD features of a given word image, first, the standardized word image $I_f(x, y)$ is thinned (skeletonised) to obtain thinned binary word image $I_t(x, y)$ whose strokes are one pixel thick. $I_t(x, y)$ is then divided into $m \times m$ cells (see figure 4.11(a)) where $m = 15$. Figure 4.11(b) is a sample $m \times m$ cell showing cell center $C(x, y)$, foreground pixels $p1(x_1, y_1)$ and $p2(x_2, y_2)$, orientations θ_1 and θ_2 respectively of $p1(x_1, y_1)$ and $p2(x_2, y_2)$ w.r.t $C(x, y)$, and displacements d_1 and d_2 respectively of foreground pixels *p1* and *p2* from cell center $C(x, y)$.

For a foreground/object pixel $(P_i(x_i, y_i))$ within the cell, its displacement (d_i) from the cell center (x, y) is obtained using equation 4.16.

$$d_i = \sqrt{(x_i - x)^2 + (y_i - y)^2} \tag{4.16}$$

Orientation (θ_i) of a foreground pixel $(P_i(x_i, y_i))$ w.r.t cell center $C(x, y)$ is computed by equation 4.17.

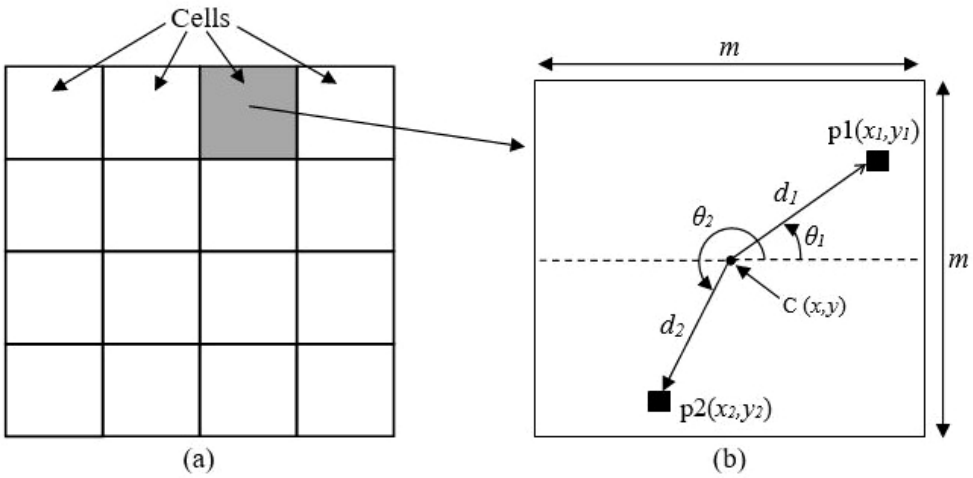


Figure 4.11: CHOD computation: (a) is a thinned word image $I_t(x, y)$ divided into $m \times m$ cells, and (b) is a $m \times m$ cell showing foreground pixels *p1* and *p2*. d_1 and d_2 are, respectively, displacements of foreground pixels p1 and p2 from cell center $C(x, y)$ whereas θ_1 and θ_2 are respectively orientations of *p1* and *p2* w.r.t $C(x, y)$

$$\theta_i\left(x_i, y_i\right) = \arctan\left(\frac{y_i - y}{x_i - x}\right) \tag{4.17}$$

Orientations are measured in counter clockwise direction from right-going horizontal axis whose origin is cell center (C) as shown in figure 4.11(b). θ_i ranges between 0 and 360 degrees. Orientation angles are then quantized into 18 bins each of bin width of 20 degrees. Therefore, each foreground pixel (p_i) in a cell is described by displacement d from cell center and orientation angle θ (or bin). Bin index (bin_i) is the quotient obtained from division of θ_i by 20 (bin width in degrees). For $\theta = 360^o$ the bin index is 17. For each cell, a 1D cell descriptor vector is obtained by voting process. In a particular cell, votes are a function of displacement (d_i) of foreground pixels from the cell center of their respective cells. That is, votes are displacement magnitudes. Each bin index receives votes of corresponding displacements which are then summed up. Votes for bins are represented as a histogram which is the cell histogram of oriented displacement (CHOD), whose length is 18. CHOD describes foreground pixels (portion of strokes) within a cell of a word image. Bin width of 20 degrees is used because it was experimentally found to be the optimum width at which shape descriptor obtained leads to best performing word spotting model.

(b) **Computing block histogram of oriented displacement (BHOD)**

To compute BHOD, adjacent cells are joined together to form overlapping blocks of 2×2 cells such that the dimension of each block is $2m \times 2m$ where m is a cell dimension fixed at 15 as explained in section 4.6.2.4. Thus, dimension of each block is 30×30 pixels. CHOD of cells in a block are concatenated in left-right, top-down order to form a single block descriptor vector referred to as block histogram of displacement (BHOD), of length 72. This is the first level-concatenation. This 1^{st} level concatenation is for 2 purposes: (i) helps to minimize intra-class variation resulting mainly from variation of handwriting styles in different writers, and (ii) enhances discriminating powers of the final descriptor. BHOD is then normalized by L2 Euclidean norm as given in equation 4.18.

$$BHOD_{norm} = \frac{BHOD}{\sqrt{|BHOD|^2 + \epsilon}} \tag{4.18}$$

Where ϵ is a very small positive regularization number (< 1) used to prevent division by zero. It is so small that it will not change the computed result.

(c) **Computation of local histogram of oriented displacement (LHOD)**

LHOD is obtained by concatenating $BHOD_{norm}$ descriptors resulting from the whole word image, in left-right, top-down order into a single descriptor vector. This is the 2^{nd} level concatenation which is global in scope.

(d) **Computation of global histogram of oriented displacement (GHOD)**

The process of computing GHOD is the same as that of CHOD (section 4.5.4.2(a)), taking the entire word image as a cell. Thus, the centre used is image centre $(w/2, h/2)$ w is width and h is height the word image. Displacements and orientation angles of foreground (text stroke) pixels are calculated with respect to global image centre $(w/2, h/2)$ and then voted to their respective angle bins. The length of GHOD vector obtained is 18 which is equal to number of bins used.

(e) **Computation of Integral Histogram of Oriented Displacement (IHOD)**

IHOD is obtained by concatenating LHOD and GHOD vectors. This is the 3^{rd} and final level concatenation resulting in formation of one combined shape descriptor called IHOD. IHOD descriptor is integral in that it captures both local and global shape information of text strokes, thus enhancing its discriminating power. IHOD descriptor is used in training MLP classifier for word spotting task.

IHOD is computed using almost similar approach as histogram of oriented displacement (HOD) descriptor by Gowayyed et al [153]. HOD [153] is used as a 2D trajectory descriptor to represent 3D motion of human body joints. In computing HOD [153], length of displacement of a body joint between two successive time steps and in a given direction is a vote in orientation angles histogram. HOD is used to give temporal behaviour of human joints. However, IHOD descriptor differs from HOD descriptor used by Gowayyed et al [153] in 6 ways: (i) IHOD captures both global and local information while HOD captures local information only, (ii) IHOD is computed using all foreground pixels while HOD is computed using human body joints (interest points) only, (iii) in IHOD, displacement is centroid distance from foreground pixels to a fixed cell centroid while for HOD, displacement is distance moved by body joint between 2 consecutive time steps, (iv) IHOD describes shape of text strokes while HOD describe temporal behaviour of body joint motion, (v) IHOD is constructed by 3-level concatenation of CHOD, BHOD, LHOD, and GHOD descriptors whereas HOD is constructed by 1-level concatenation of orientation angle histograms of human body joints, and (vi) IHOD is spatial whereas HOD is spatio-temporal.

4.5.4.3 IHOD vector size

For a word image of width w and height h, $cell\_dim = m \times m$, $block\_dim = 2m \times 2m$, the number of blocks in horizontal axis (B_H) and vertical axis (B_V) is given by equations 4.19 and 4.20 respectively.

$$B_H = \frac{w - 2 * m}{s} + 1 \tag{4.19}$$

$$B_V = \frac{h - 2 * m}{s} + 1 \tag{4.20}$$

Where s is stride and $m = 15$ is cell dimension fixed as explained in section 4.6.2.4. Stride equal to cell dimension (m) is used. The length of LHOD descriptor (section 4.5.4.2 (c)) is given by equation 4.21.

$$LHOD_{size} = 72 * B_H * B_V \tag{4.21}$$

Since GHOD size is 18 as explained in section 4.5.4.2(d), length of IHOD vector is given by equation 4.22.

$$IHOD_{size} = LHOD_{size} + GHOD_{size} = 72 * B_H * B_V + 18 \tag{4.22}$$

For a word image of size 45×135, $m = 15$, and $s = 15$ pixels, IHOD size is 1170.

4.5.5 Training

MLP-based HWS model is developed by training MLP classifier using IHOD descriptor. Python 3.6 was used together with Keras module. Keras sequential model is used where it is constructed by stacking fully connected layers in a linear manner [154]. Training was carried out using a core i5 8^{th} generation HP laptop with 1TB HDD, 8GB RAM, and 1.8GHz clock frequency. Figure 4.12 shows architecture of a fully connected MLP network consisting of 3 layers that was trained so as to obtain HWS model. The input layer (1^{st} layer) has 4,096 units, 2^{nd} layer has 2,048 units, and the 3^{rd} (last/output) layer has k units where k is number of classes. Rectified linear unit (ReLU) is the activation function used for layers 1 and 2. The activation function used in layer 3 (output layer) is Softmax. Optimizer used is adaptive moment (Adam) [123]. Categorical cross-entropy is the loss function used because it is suitable for multiple classification tasks. It is given by equation 4.23. Drop out of 30% was used as regularization to reduce effect of overfitting. Batch training was done with 15 epochs and batch size of 64 samples.

$$Loss = -\sum_{i=1}^{k} t_i log\ y_i \tag{4.23}$$

Where $i = 0, 1, 2, ..., k$ is index of input training sample, k is number of classes, t is actual output label, and y is predicted output of the network.

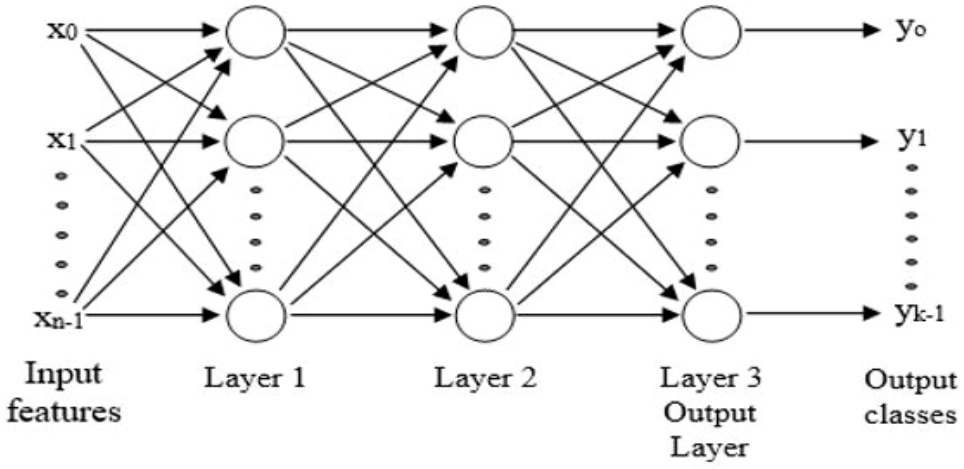


Figure 4.12: Fully connected MLP network architecture used to develop handwritten word spotting model, where k is number of output classes and n is input feature size

The proposed technique has the following merits:

- ☛ IHOD features are easy to compute.
- ☛ The IHOD shape descriptor used has high discriminating power thus explaining the high performance of the proposed technique. This is due to the descriptor capturing both local and global information of word and character strokes. Local information is obtained from local (cell and block)-based computation of histogram of oriented displacement (LHOD features) whereas global information is obtained from global-based computation of histogram of oriented displacement (GHOD features).
- ☛ The proposed technique has low computational cost. In computation of IHOD features, the only calculations involved are those of angles, local/global oriented displacements, and voting by bin-wise accumulation of lengths of displacements.

☛ During model training, only a 3-layer fully connected network is used which is less computational and need fewer training epochs compared to CNN-based techniques which need high number of training epochs. This makes the proposed technique memory efficient.

4.6 Experimental results and discussion

4.6.1 Evaluation of CTA technique for word segmentation

The publicly available International Conference on Document Analysis and Recognition (ICDAR) 2009 and 2013 datasets [155, 156, 157] has been used to evaluate performance of CTA segmentation technique. The datasets contain scanned handwritten document images. These handwritten documents consist of texts in different languages like Germany, French, English, and Indic (like Devanagari, Bengali, Telugu etc. of Indian origin). Output performance is objectively evaluated using metrics like recognition accuracy (RA), detection rate (DR), and performance metric (PM). Match score (MS) is a measure of matching between test segmented word and ground truth based on intersection of foreground pixels on both word images for a given region. It is computed using equation 4.24 [155, 156].

$$MS(i,j) = \frac{T\left(G_j \cap R_i\right) \cap I}{T\left(G_j \cup R_i\right) \cap I} \tag{4.24}$$

Where R_i are text pixels in i^{th} test region, G_j are text pixels in j^{th} ground truth region, I represent pixels of entire text document image, and $T(.)$ is a function that counts text pixels. DR is proportion of one-to-one (o2o) matches to number of ground truth elements (words/lines). RA is proportion of o2o matches to number of test elements (words in this case). DR, RA, and PM are computed respectively by equations 4.25-4.27 [155, 156].

$$DR = \frac{o2o}{N} \tag{4.25}$$

$$RA = \frac{o2o}{M} \tag{4.26}$$

$$PM = \frac{2DR \times RA}{DR + RA} \tag{4.27}$$

Where M and N are respectively numbers of test and ground truth elements (words). Large values of RA, DR, PM is indicative of a better segmentation technique.

For ICDAR2009 dataset [155, 156], CTA segmentation technique obtained overall scores of 98.22%, 98.56%, and 97.89% respectively for PM, DR, and RA. For ICDAR2013 dataset [157], the CTA segmentation technique attained overall scores of 98.58%, 98.02%, and 99.14% respectively for PM, RA, and DR metrics. The CTA segmentation method performs well due to good modelling of inter-CC gap distances that effectively identifies and distinguishes each of them as either inter or intra word gaps. Also, the JBA method (sections 4.4.4) contributes immensely to the good performance since it effectively segments crossing words especially those whose strokes extend into regions or 'territories' of neighbouring words and crossing with their (neighbouring words) strokes. For crossing words, each of the crossing strokes or strokes straying to regions of other words is traced and joined to the word it duly belongs. JBA method also effectively segments/separates touching words as afore-explained in sections 4.4.4 and 4.4.5. Figure 4.13 shows results for CTA word segmentation technique for handwritten text blocks obtained from ICDAR2009 [155, 156]. In the figure, 1^{st} row consists of Latin text blocks while rows 2 & 3 consists of non-Latin text blocks. From figure 4.13, it can generally be seen that very good segmentation output is obtained especially in overlapping and crossing words. Overlapping, touching, and crossing words are efficiently segmented because of the novel JBA method (sections 4.4.4) that associates well the junction branches to their parent core word segments. This is where many of the segmentation techniques in literature perform dismally.

It can be seen that the proposed CTA segmentation technique is effective in segmenting crossing and overlapping words as shown figure 4.13. This is because JBA method (section 4.4.4) used is effective in segmenting crossing words where other methods fail.

Figure 4.13: Handwritten word segmentation results by CTA technique

Each of the ICDAR2009 [155, 156] and ICDAR2013 [157] datasets were divided into 2 main categories based on the language of the handwritten texts. The categories are: Latin/Latin-like texts (LLT) and non-Latin/Latin-like texts (non-LLT). Examples of LLT are those scripts that use Latin or Latin-like alphabets which includes English, French, Greek whereas non-LLT are those texts whose structure of alphabets/characters are very different from Latin which includes Indic (like Bengali, Telugu, Kanada), Chinese, Arabic among others. Figures 4.14 shows performance of the CTA word segmentation method for non-LLT and LLT categories for ICDAR2009 [155, 156], and ICDAR2013 [157] datasets.

It can be seen that CTA segmentation technique has slightly better performance for LLT category as compared to non-LLT category for both ICDAR2009 [155, 156], and ICDAR2013 [157] datasets. This is because presence of many diacritic symbols/marks in non-LLT texts makes full word segmentation a bit challenging especially in cases where diacritic symbols/marks touch or are near core segment of a target word. This lowers segmentation performance for non-LLT HWDs compared to LLT HWDs as seen figure 4.14(a & b). In LLT

HWDs, there are not many diacritic symbols/marks compared to non-LLT HWDs. Thus, full word segmentation in LLT HWDs is less challenging compared to non-LLT HWDs.

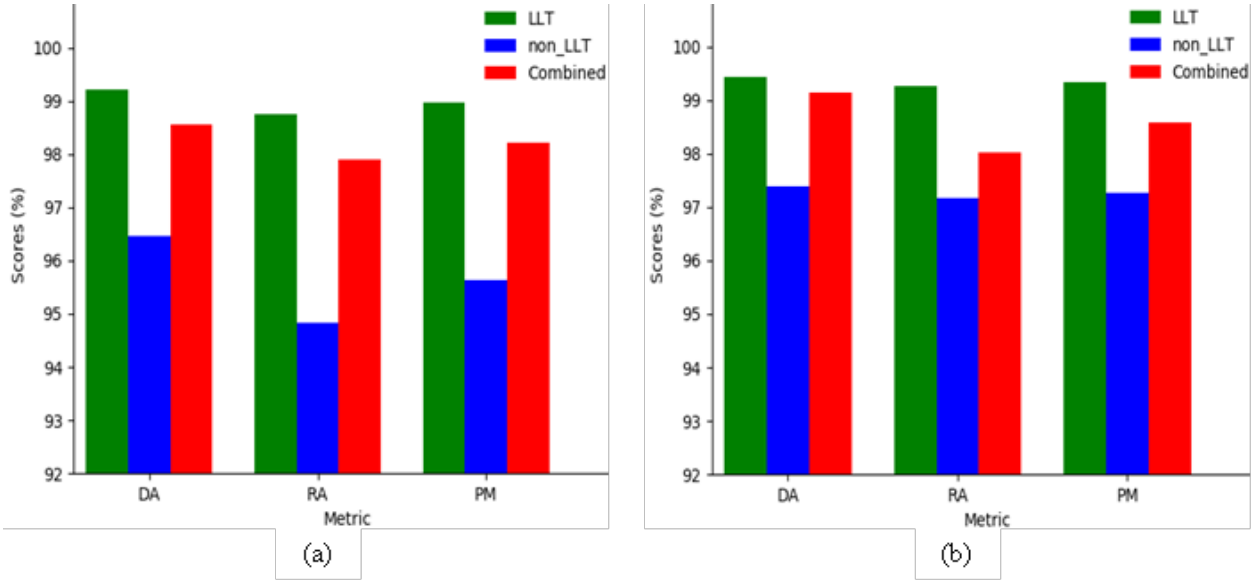


Figure 4.14: Performance of CTA segmentation technique for non-LLT and LLT categories for (a) ICDAR2009 [155, 156], and (b) ICDAR2013 [157] datasets

Performance of CTA word segmentation method is also compared with state-of-the-art segmentation methods using performance metric (PM), recognition accuracy (RA), and detection rate (DR) metrics for both ICDAR2009 [155, 156] and ICDAR2013 [157] data sets as shown in table 4.1. Performance metrics for segmentation techniques by Chaudhuri and Pal [158], Karmakar et al [159], Jain and Singh [160], Sharma and Dhaka [161], and Dahake et al [162] were obtained from the work of Sharma and Dhaka [163]. For run length smoothing algorithm (RLSA) for word segmentation [164], its performance metrics have been obtained from its implementation by Gatos et al [155]. It can be seen that CTA word segmentation method has the best performance when compared with other methods for both ICDAR2009 [155, 156] and ICDAR2013 [157] data sets as shown in table 4.1. This shows that the proposed CTA technique is efficient and robust. For ICDAR2009 [155, 156] data set for DR, RA, and PM metrics, ILSP-LWSeg-09 [155] and Jain and Singh [160] rank 2^{nd} and 3^{rd} respectively while RLSA [164] came last. For ICDAR2013 [157] data set, Sharma and Dhaka [161] and Jain and Singh [160] comes 2^{nd} and 3^{rd} respectively for DR and PM metrics scores whereas Karmakar et al. [159] is last for the same metrics. For RA metric, Jain and Singh

(2014) and Sharma and Dhaka [161] rank 2^{nd} and 3^{rd} respectively as Karmakar et al.[159] comes last.

Comparison of performance of CTA segmentation method with state-of-the-art segmentation methods for ICDAR2009 [155, 156] and ICDAR2013 [157]

Table 4.1: Comparison of performance of CTA segmentation method with state-of-the-art segmentation methods for ICDAR2009 [155, 156] and ICDAR2013 [157]

Method	ICDAR2009 [155, 156]			ICDAR2013 [157]		
	DR(%)	RA(%)	PM(%)	DR(%)	RA(%)	PM(%)
ILSP-LWSeg-09 [155]	95.16	94.38	94.77	-	-	-
Chaudhuri and Pal [158]	83.55	89.29	90.71	90.62	89.45	89.96
Karmakar et al [159]	87.86	86.91	89.62	89.62	84.45	86.96
Sharma & Dhaka [161]	87.93	88.37	88.15	94.37	94.38	94.77
Jain and Singh [160]	90.50	91.55	91.03	94.25	94.98	94.61
Dahake et al [162]	87.82	91.85	83.16	92.77	92.99	91.68
Sharma and Dhaka [163]	96.32	95.74	95.72	98.32	96.74	95.99
RLSA [164]	80.78	77.68	79.20	-	-	-
Jindal & Jindal [165]	88.31	90.98	89.62	93.67	93.98	93.78
Proposed	98.56	97.89	98.22	99.14	98.021	98.58

In order to evaluate how CTA word segmentation method performs specifically in segmenting crossing words, a set of 400 crossing words (300 of LLT and 100 of non-LLT category) cropped from HWDs of FireMaker dataset [166], ICDAR2009 [155, 156], and ICDAR2013 [157] datasets were used. Figure 4.15a shows various forms of crossing present in test word images used to evaluate the proposed CTA word segmentation method. Figure 4.15a rows (i-iv) shows desirable segmentation results of CTA word segmentation method for crossing words. This is because CTA word segmentation method is able to discriminatively trace strokes/portions of a word that extends into regions of adjacent words and crossing with their strokes.

Table 4.2 and figure 4.15b shows performance of CTA word segmentation method in segmenting crossing words. From table 4.2, it can be seen that 96.7% of LLT crossing words are correctly segmented while 98% of non-LLT crossing words are correctly segmented. When the LLT and non-LLT crossing words are combined, the segmentation accuracy is 97%. The high segmentation accuracy shows effectiveness of CTA technique in addressing the challenge

of crossing words where other methods fail. From figure 4.15b, it can be seen that segmentation performance of CTA method for non-Latin crossing word images is better than that for Latin crossing word images. This is because non-LLT crossing strokes/words are less complex compared to LLT crossing strokes/words. Consequently, the proposed CTA segmentation technique does not segment well exceptional LLT crossings like in figure 4.15a row v. CTA word segmentation method could not segment well only in few cases especially where stroke of adjacent words fit into each other so well that a stroke segment of a word is erroneously assigned to another word as seen in figure 4.15a row v.

Table 4.2: Performance of CTA word segmentation method in segmenting crossing words

	Word-pairs	Correct segmentations	Accuracy (%)
Non-LLT	100	98	98.0
LLT	300	290	96.7
Combined	400	388	97.0

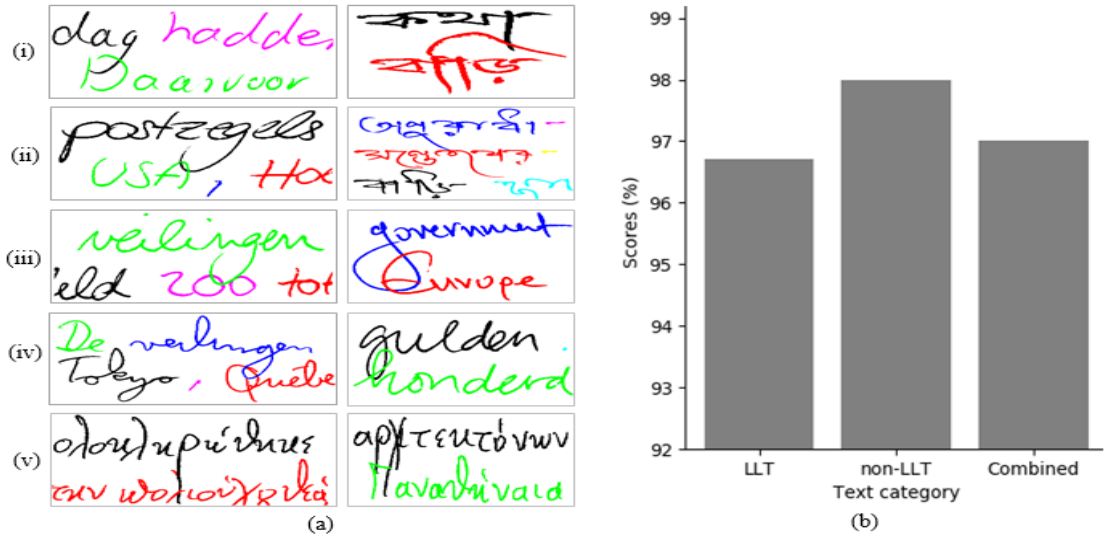


Figure 4.15: Word segmentation results by CTA word segmentation method for crossing words, (a) sample crossing words segmented, and (b) graphical representation of segmentation performance of CTA word segmentation method

4.6.2 Evaluation of handwritten Word spotting (HWS) model

4.6.2.1 Evaluation Datasets

The MLP-based HWS technique has been evaluated with IIIT-HW-DEV [167] and RoyDB Devanagari [39] datasets that are publicly available. IIIT-HW-DEV dataset [167] has 95,381 instances of words written by 12 different writers. The data set has 9540 unique words. It is divided into 3 sets: train (70%), validation (15%), and testing (15%) sets. RoyDB Devanagari dataset [39] has a lexicon size of 1,957 unique words, and a total of 16,128 instances of handwritten Hindi word images. It is divided into train set (10,667 words), testing set (3,589 words), and validation set (1,872 words).

4.6.2.2 Data augmentation

Machine learning-based models need a lot of training data for development of efficient models. In cases where available data is insufficient, data augmentation is used to supplement the already provided data for development of robust and efficient models. Data augmentation is the manipulation of real data provided using some techniques so as to generate another data. The generated data and real data have many similarities. The augmented data together with real data help to reduce over-fitting and hence enhance robustness of the trained models. The evaluation datasets (section 4.6.2.1) to be used is not sufficient for developing trained models from scratch, thus data augmentation is performed to generate extra data. In this work, the augmented techniques used are elastic distortion [168] and affine transformation. In affine transformation, the approaches used are full and partial shearing, and scaling. In partial shearing, only half of the word image is subjected to shearing. After augmentation, a data set is increased 10-15 times. Samples of augmented word images are shown in figure 4.16 obtained by performing affine transformation on original word images in column 0. It can be seen that good augmented images (figure 4.16 columns 1-4) are obtained that have similar structural characteristics to their respective original images in column 0. These augmented word images are used to supplement available images in model development.

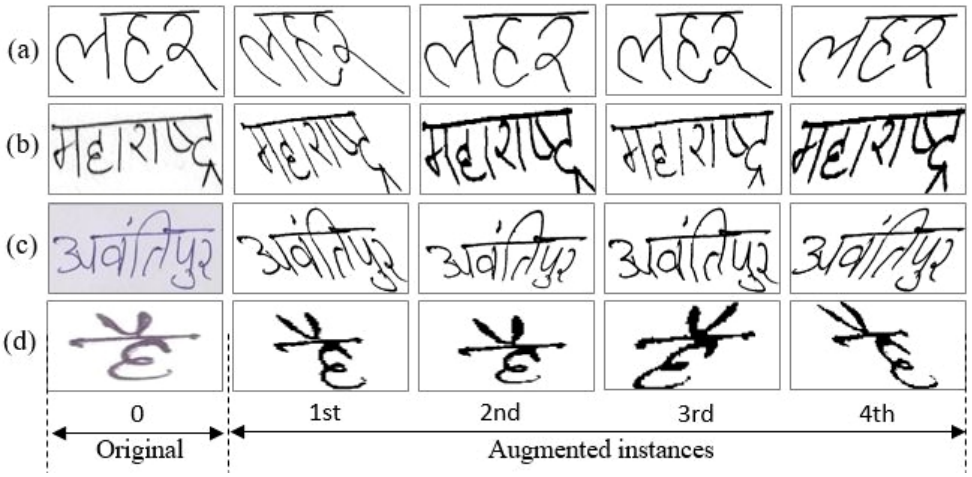


Figure 4.16: Sample instances of augmented word images in columns 1-4 (a-d) obtained by applying affine transformation and elastic distortion [168] on respective original images in columns 0 (a-d). Original image in row (a) is from RoyB dataset [39] whereas for rows b and c, original images are from IIIT-HW-DEV dataset [167]

4.6.2.3 Evaluation metrics

In evaluating HWS technique, mean average precision (MAP) and k-precision (kPr) metrics are used.

(a) **k-precision (kPr)**

This is the proportion of relevant items/words retrieved in top k retrieved words/items. Equation 4.28 is used to compute kPr metric.

$$kPr = \frac{k_o}{k} \tag{4.28}$$

Where k_o is number of relevant words in top k retrieved words.

(b) **Mean average precision (MAP)**

MAP is the average precision in increasing retrieving size (rank). It is computed using equation 4.29 [38].

$$MAP = \frac{\sum_{k=1}^{j} (kPr \times rel\,(k))}{No.\ of\ relevant\ words} \tag{4.29}$$

Where $k = 0, 1, 2, ..., j$ is index of retrieved word, j is the number of similar word images retrieved and $rel(k)$ is obtained using equation 4.30.

$$\text{rel}(k) = \begin{cases} 0 & \text{if word at rank } k \text{ is not relevant} \\ \\ 1 & \text{if word at rank } k \text{ is relevant} \end{cases} \tag{4.30}$$

Both MAP and kPr metrics are in the range $0 - 1$. Large values of kPr and MAP metrics indicates that the proposed HWS technique is efficient. MAP is a good metric for evaluating performance of retrieving systems [169] because it shows retrieval size at which performance of a word spotting system is optimum.

4.6.2.4 Evaluation of handwritten Word spotting (HWS) model

The MLP-based HWS technique is evaluated using query by example (QBE) method to search instances of same words from a database of candidate word images. The MLP-based HWS technique is evaluated on the two afore-mentioned data sets (section 4.7.1): IIIT-HW-DEV [167], and RoyDB Devanagari [39] datasets. First, performance of the proposed HWS technique is evaluated when different cell sizes are used during feature extraction (discussed in section 4.5.4).

Table 4.3 and figure 4.17 shows evaluation scores of MAP and kPr metrics for the MLP-based HWS technique for various cell sizes ($9 - 23$ pixels) used during IHOD feature extraction. It is seen from table 4.3 and figure 4.17 that MAP and kPr scores increase with increase in cell size to maximum when cell size is 15 pixels, beyond which the performance starts to fall gradually. When cell sizes of above 15 pixels are used during feature extraction, much useful information is lost due to lumping together of large foreground regions. This leads to less efficient models being obtained, thus reduced performance. When cell size used in features extraction is less than 15 pixels, many redundant features are obtained that lead to increased feature dimension. This causes network complexity, hence less efficient model obtained during training. The most efficient model is obtained when cell sizes of dimensions (width and height) of 15 pixels are used during feature extraction (section 4.5.4), when

feature redundancy is at minimum.

Table 4.3: Performance of MLP-based word spotting technique for various cell dimensions ($m \times m$) used during IHOD feature extraction

m	9	11	13	**15**	17	19	21	23
kPr	0.7713	0.7887	0.8475	**0.9758**	0.8671	0.8498	0.8423	0.8151
MAP	0.7435	0.7611	0.7971	**0.9736**	8411	0.8250	0.8151	0.7891

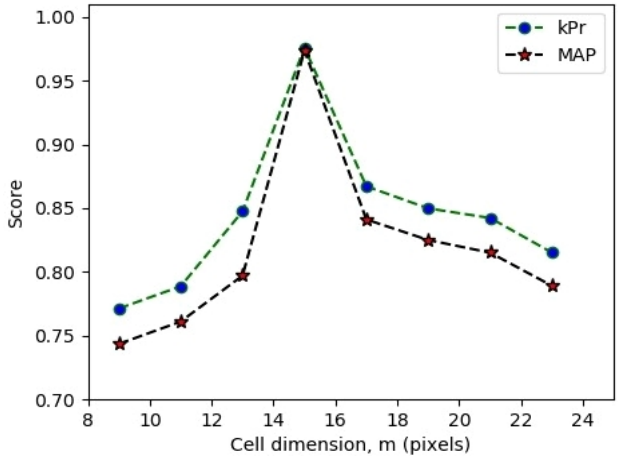


Figure 4.17: MLP-based word spotting model performance plot for various cell sizes used during IHOD feature extraction, where kPr is k-precision, MAP is mean average precision, and m is cell dimension in pixels

In evaluating MLP-based HWS technique with IIIT-HW-DEV dataset [167], query images used are from test set. It was noted that all unique words from query set (test set) have frequency of 3 and below except one word image with frequency of 4. Therefore, when computing kPr and MAP metrics, 3 retrievals were used. Figure 4.18 shows samples of retrieved instances of respective query images from IIIT-HW-DEV dataset [167]. It can be seen that the top 3 retrieved instances are similar to the query word image whereas 4^{th} and 5^{th} retrieved instances are not. This is because almost all query word images have frequency of 3 and below as already mentioned. This shows that the proposed HWS technique is efficient in retrieving similar words from a database for a given query word image.

Table 4.4 shows comparison of performance of the proposed MLP-based HWS method with that of state-of-the-art word spotting methods on IIIT-HW-DEV [167] and RoyDB [39] Devanagari datasets. Performance scores for end2end network [170] are obtained from the

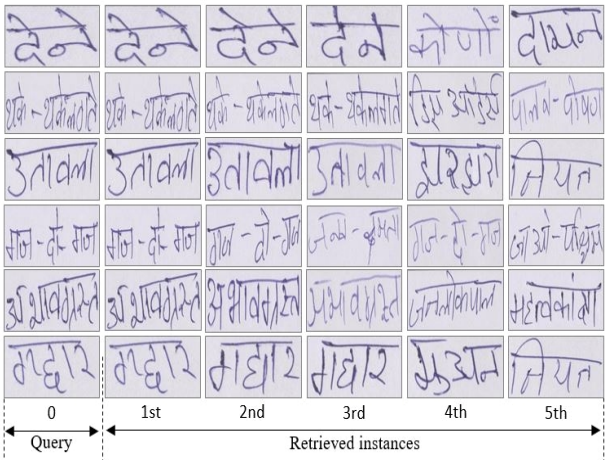


Figure 4.18: Retrieved word images of IIIT-HW-DEV dataset [167], using MLP-based HWS technique where column 0 consists of query images and columns $1-5$ has retrieved instances of words in descending order of scores

work of Dutta et al [142]. From evaluation of the MLP-based word spotting technique in table 4.4, it can be seen that MAP and k-precision metrics are very high. Performance of MLP-based HWS method compares very well with that of End2End Network [170] technique for IIIT-HW-DEV dataset [167].

Though End2End Network [170] performs slightly better than MLP-based HWS technique by a very small margin, it is computationally expensive and relies exclusively on GPU. In our case, training of MLP classifier (section 4.5.5) with IIIT-HW-DEV dataset [167] to obtain word spotting model was done using a laptop as noted before. Training took 30 minutes, for 15 epochs. That is 2 minutes per epoch which is compatible with a variety of portable digital devices. For RoyDB dataset [39], the proposed method has the best performance. Training with RoyDB dataset [39] took 4 minutes for 15 epochs translating to about 16 seconds per epoch. This means that the MLP-based word spotting technique is computationally less expensive and posts appreciable performance. For RoyDB dataset [39], it can be noted that kPr and MAP scores are higher than those for IIIT-HW-DEV dataset [167]. This is because IIIT-HW-DEV dataset [167] has larger lexicon (9,540) as compared to RoyDB dataset [39] with lexicon of 1,957 words. As the lexicon increases, there are many classes whose handwritten structure are similar, hence higher chances of misclassification.

Figures 4.19 and 4.20 respectively show k-precision and MAP scores of the proposed HWS

Table 4.4: Comparison of performance of HWS method with state-of-the-art word spotting techniques for RoyDB [39] and IIIT-HW-Dev [167] datasets

Technique	IIIT-HW-Dev [167]		RoyDB [39]	
	kPr	MAP	kPr	MAP
End2End Network [170]	-	**0.6387**	-	0.9439
HMM-based method [39]	-	-	0.9495	-
Proposed	**0.6426**	0.6195	**0.9631**	**0.9613**

technique for different top k words retrieved for various datasets. It can be seen that for RoyDB [39] dataset, the performance increases as number of top words retrieved increases up to the best scores at 5 top words retrieved. For IIIT-HW-DEV dataset [167], the performance peaks at 3 top words beyond which it starts to decrease. This is because the unique words in the candidate(test) set have frequencies of 3 and below except one word image which has a frequency of 4. Therefore, when the number of top words retrieved is beyond 3, dissimilar instances are included which accounts for the decreased scores of kPr and MAP

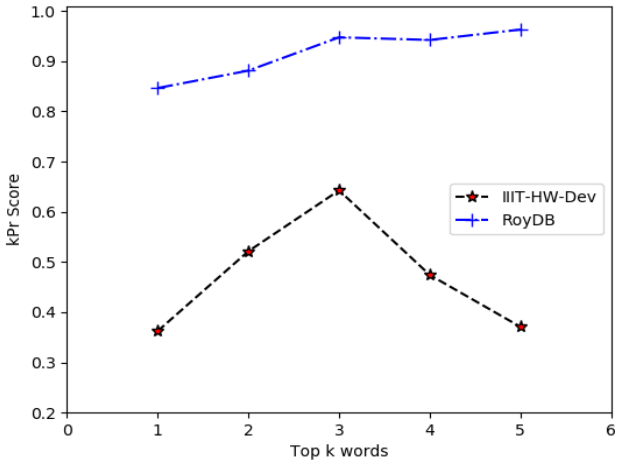


Figure 4.19: Precision of the MLP-based word spotting technique on k top words retrieved from RoyDB [39] and IIIT-HW-DEV [167] datasets

4.7 Conclusion

A segmentation-based MLP-based handwritten word spotting technique has been presented in this chapter. The word spotting technique uses IHOD feature descriptor to train MLP

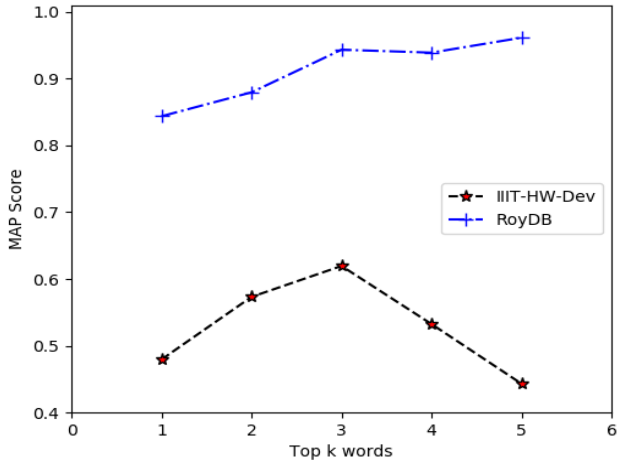


Figure 4.20: MAP of the MLP-based word spotting technique on k top words retrieved from RoyDB [39] and IIIT-HW-DEV [167] datasets

classifier for handwritten word spotting task. IHOD feature descriptor has high discriminating power, easy to compute, has low computational cost, and has very good performance. IHOD has both local and global information making it to have high discriminating power and robust in performance. The MLP-based word spotting technique is very efficient and has good performance compared to state-of-the-the-art word spotting techniques, and is applicable in ordinary personal computers (PCs), laptops, and PDAs (personal digital assistants). An efficient CTA technique for word segmentation has also been presented. The technique performs well in segmentation of overlapping and crossing words without segmentation errors like under/over-segmentation.

Chapter 5

LANGUAGE IDENTIFICATION FOR HISTORICAL MANUSCRIPT DOCUMENTS

5.1 Introduction

This chapter presents a model-based technique for identifying language of textual content of historical manuscript images (HMI). When we come across a new historical manuscript document, the first interest is to identify the language of its textual contents. Knowing the language of a historical manuscript makes it easier to perform other manuscript management tasks like identifying the manuscript's geographical source, author, era or production time, appropriate translator among others. Knowing the language will also help in appropriate indexing for storage and future retrieval purposes.

Language identification (LID) model is used to identify language of a text or manuscript. The LID model is obtained by training long short-term memory (LSTM) recurrent neural network (RNN). LID is one of the crucial and main tasks associated with management of historical manuscripts. The input to the proposed LID system is text word in ASCII form which can be modelled well using LSTM. Word image cannot be used because from it, only shape (or textural) features can be extracted. Such an approach is not feasible especially

for languages that are written with same script like languages written using Latin script of alphabets like English, French, Spanish, and Swahili (from East Africa). It is challenging to distinguish such languages using shape or texture features since they are similar.

5.2 Language Identification (LID)

Language identification (LID) is a task of determining natural language of a written word, sign, or any other spoken form of communication [171, 172]. For the case of written text, it can be a word, phrase, sentence, paragraph or full page. It is a primary step in any natural language processing system (NLP) [173]. Language has a major role as a main medium for communication, transmitting culture across generations, acquisition of knowledge. It is associated with communities or groups of people, culture, geographical area, grammar, and history.

Deep learning approaches have proved promising for LID task due to their desirable performances. These approaches include RNN, gated recurrent unit (GRU) [174], LSTM [99, 175, 176], and CNN (convolutional neural network). GRU and LSTM are variants of RNN. Of the mentioned approaches, LSTM based LID methods performs best. LSTM has long term dependencies that is very valuable when dealing with sequential data like spoken or written text. Standard RNN consists of recurrent layers in which output of one layer in a given time step is the input for the next layer during next time step. Standard RNN has 4 main drawbacks: (i) inability to store information for a long duration, (ii) inability to control information to be discarded (forgotten) or retained (remembered), (iii) vanishing and exploding gradients during training process, and (iv) it processes information from previous time steps only especially for unidirectional RNN [176, 177]. LSTM is a variant of RNN with LSTM cells/units in its hidden layers. It was pioneered and designed by Hochreiter and Schmidhuber [99] purposely to mitigate the aforementioned shortcomings of standard RNN specially in remembering information for long periods of time and overcoming the problem of vanishing and exploding gradients. The LSTM cells have various components called gates that control the information to be discarded, retained and output to the next

LSTM unit [178]. These gates are input, output, and forget gates. The output from LSTM is dependent on information from distant past time step [99].

5.3 Problem statement

LID being a crucial manuscript management tasks, efficient and robust LID techniques are necessary. N-gram based methods have often been used for LID tasks, which have high computational cost especially when train size and number of language-classes is large. Also, the N-gram methods are not efficient especially for closely/semantically related languages. Therefore, efficient LID methods are needed that are also capable of distinguishing similar languages.

5.4 Fragmented LSTM for Language Identification (Frag-LSTM-LID)

In LID task, fragmented LSTM is used. This is because from literature, long term dependencies in LSTM models well sequential data like text, spoken speech, stock prices over a long period of time among others. It consists of 3 main steps: fragmentation, pre-processing (PP), and LSTM process as shown in figure 5.1 that shows the framework of Frag-LSTM-LID. The input for the proposed Frag-LSTM-LID framework is a full text string (word) whose length is greater than 5 characters. The word-size should be greater than 5 for appropriate fragments to be obtained. The steps in Frag-LSTM-LID are discussed in detail in sections that follow.

5.4.1 Fragmentation

In this step, input word is subdivided to 2 equal or almost equal non-overlapping sub-words. Sub-division is done such that character order in original word is retained in sub-words. The sub-words are also referred to as fragmented words (fragWords). For example, if the input word “COMING”, it is subdivided to “COM” and “ING” sub-words/fragWords.

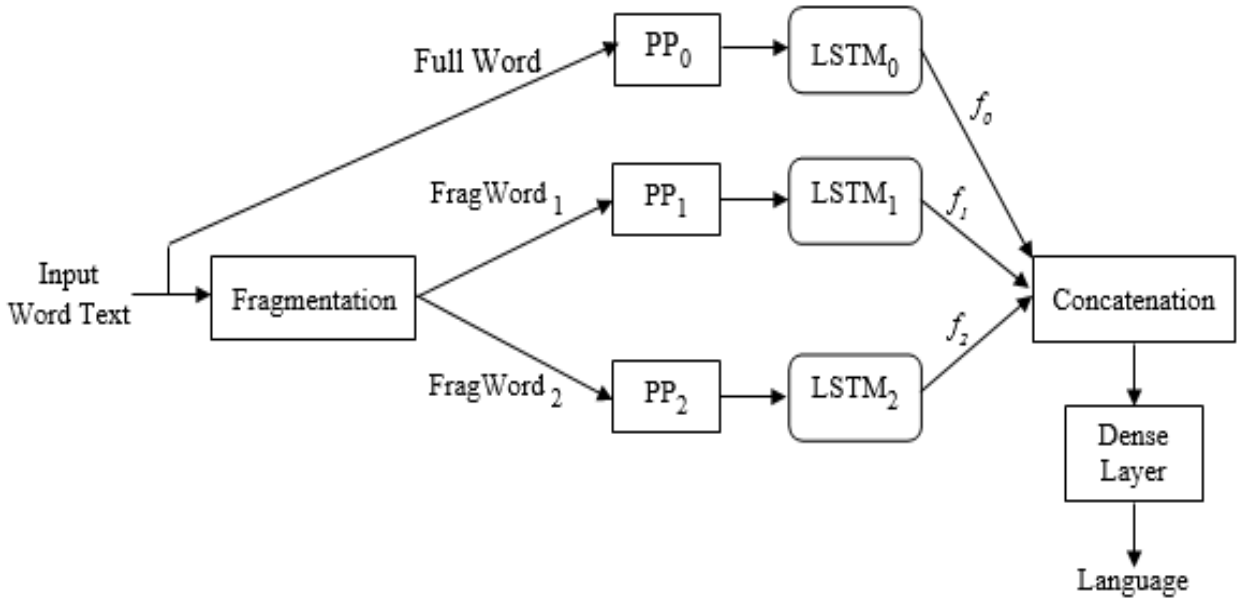


Figure 5.1: Framework of Frag-LSTM-LID where FragWord denotes fragmented word or sub-word, PP means pre-processing, LSTM means long short-term memory, and f_i denotes outputs of LSTM networks

5.4.2 Pre-processing

In this step, the sub-words from previous step undergo character tokenization and word embedding processes. The output is a numerical vector for each sub-word, which is used in training a LSTM network.

(a) **Character tokenization**

This is a process of converting letters/characters of a word to index values consisting of integer numbers. The output is an integer sequence. A character tokenizer is created using a dictionary of letters/characters obtained from train data. The dictionary of characters is a set of unique letters in all words in train set. Some non-alphabets like apostrophes (‘) are included in the dictionary. All other punctuation marks are not included in the dictionary. Each character in the dictionary is assigned a unique integer starting from 1. Zero (0) is preserved for out of dictionary characters during testing phase. The tokenizer is then used to convert characters in a word into sequential data where each character/letter is represented with numerical index. Padding and truncating are used to set sequential data obtained to same length. Padding is done for

data with length smaller than the required one. Truncating is done for data with length larger than the required one where the extra characters are discarded. Optimized length of the sequential data is computed using equation 5.1 [179].

$$length \;=\; mean\left(len\left(w_i\right)\right) + 2 \times std\left(len\left(w_i\right)\right) \tag{5.1}$$

Where $len\left(w_i\right)$ is length of words in train dataset, and std is standard deviation of lengths of words in train dataset.

Example: Let's index Latin alphabets (A-Z) starting from 1 such that $A = 1$, $B = 2$, ..., $Y = 25$, $Z = 26$. The 6-letter word 'COMING' will be tokenized as, $[3, 15, 13, 9, 14, 7]$. If the required vector size is 6, the vector representation obtained will remain. If the required vector size is 8 (i.e., greater than length of input word), the obtained raw vector will be pre-padded with zeros such as to obtain vector size of 8 as follows: $[0, 0, 3, 15, 13, 9, 14, 7]$. If vector size required is 4 (i.e., less than length of input word), the obtained raw vector is truncated by discarding some vector elements from the end such that to obtain the required size as follows: $[3, 15, 13, 9]$.

(b) **Word embedding**

This is a technique proposed by Pennington et al [180] where a text word is represented in fixed length vector of real numbers for the purpose of text analysis. During this process of converting text word to real integer form, integer sequence of characters of a text word (section 5.4.2a) is used. The vector represents projection of characters of a word into continuous vector space such that the inter-character distance is related to their semantic and syntactic similarity and quality. The vector representation of characters of a word is based on characters in neighbourhood of a character when it is used. The commonly used pre-trained word embedding methods are Glove (Global Vectors)[180] and Word2Vec [181, 182].

In the proposed system, Kera's embedding layer is used for word embedding where the inputs (to embedding layer) are tokenized vector forms of input words as explained in section 5.4.2(a). This embedding layer is initialized with random weights to vectorize

for all characters of words in training set. The embedded word vector outputs of embedding layer are in turn used as features to train LSTM for LID task.

(c) **LSTM step**

The vector outputs of embedding layer (for each full word and sub-words of input layer) are used to train individual LSTM network. LSTM is used since it models well sequential data like that of text word or sub-word.

5.4.3 Working mechanism of LSTM

The basic unit of LSTM is LSTM cell which is composed of gates that control information it remembers or forgets. In LSTM application, information passes through many LSTM cells. Output of last LSTM cell is dense layer for final output. Gates consist of logistic sigmoid and tanh layers. These gates interact with one another to produce cell output and cell state. In addition, the gates filter the information needed from previous LSTM cell, determines the information to be retained by the cell, and the information passed on to the next cell. The architecture of LSTM cell is shown in Figure 5.2 and consists of 3 main gates: forget, input, and output gates. LSTM cell is fed by 3 data inputs: (i) input data at current time step (x_t),(ii) cell state (memory) of previous cell (C_{t-1}), and (iii) output (hidden state) of previous cell (h_{t-1}). The outputs of the LSTM cell are current hidden state (h_t) and updated cell state (C_t). The main gates of the LSTM cell are discussed in detail as follows:

(i) **Forget gate**

This gate is composed of logistic sigmoid layer. It is used to control amount of information from previous cell (C_{t-1}) that forms part of current cell state (C_t), and cell output (h_t). The gate has 3 inputs; C_{t-1}, h_{t-1}, and x_t. First, h_{t-1}, and x_t are passed through sigmoid function (σ) such that output f_t is obtained as shown in equation 5.2. f_t ranges from 0-1, where 0 means complete forgetting (discarding) of information from previous LSTM cell and 1 denotes retaining/remembering of all information from previous LSTM cell.

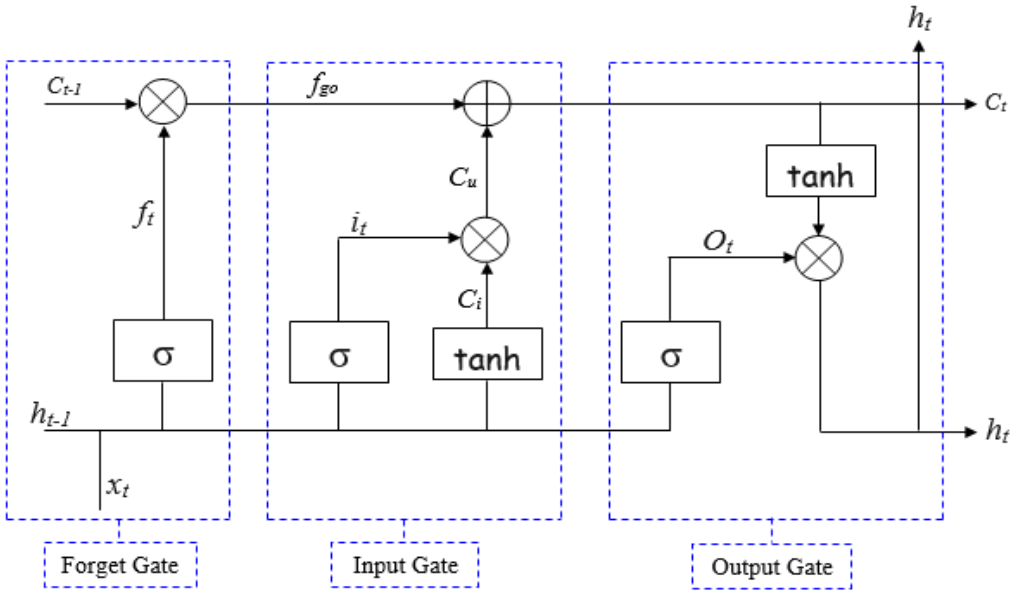


Figure 5.2: Architecture of LSTM cell where x_t is current input data point, c_t is new updated memory (cell state), c_{t-1} is memory (cell state) from previous LSTM unit, h_t is current LSTM output (or current hidden state), h_{t-1} is output (hidden state) of previous LSTM unit, f_t is output of sigmoid layer, f_{go} is forget gate output, C_u is useful information to be stored in current cell state, σ is sigmoid layer, tanh is tanh layer, $\oplus$ is adding information, and $\otimes$ is scaling of information

$$f_t = f\left(h_{t-1}, x_t\right) \tag{5.2}$$

Where f is sigmoid activation function and h_{t-1} and x_t assumes meanings as before. f_t determines the amount of useful information from previous cell (C_{t-1}) that is passed to the current cell state. Output of forget gate (f_{go}) is obtained as point-wise multiplication of f_t and C_{t-1} as shown in equation 5.3.

$$f_{go} = f_t \otimes c_{t-1} \tag{5.3}$$

Where f_t is output of sigmoid layer (equation 5.2), $\otimes$ denotes pointwise vector multiplication.

It should be noted that f_{go} is useful information from output of previous LSTM cell (i.e., C_{t-1}) that is remembered and passed on to current LSTM cell. The unremembered

information is discarded.

(ii) **Input gate**

Input gate consists of sigmoid and tanh layers. It has 3 inputs: f_{go}, h_{t-1} and x_t. This gate determines the useful and new information to be stored in current LSTM cell. Using inputs h_{t-1} and x_t, the sigmoid layer controls (selects) the information (i_t) updated as given by equation 5.4. Tanh layer is used to create a vector consisting of new candidate values (C_i) to be included in the current LSTM cell state. C_i ranges from -1 to $+1$ and it consists of all possible values of inputs (h_{t-1} and x_t) as shown in equation 5.5. Useful information (C_u) is obtained as a pointwise multiplication of i_t and C_i as shown in equation 5.6. C_u is the useful information to be added to the current LSTM cell state. Current LSTM state (C_t) is obtained as point-wise addition of C_u and f_{go} as shown in equation 5.7. C_t is also the output of input gate or new updated cell memory.

$$i_t = \sigma\left(W_i \bullet [h_{t-1}, x_t] + b_i\right) \tag{5.4}$$

$$C_i = tanh\left(W_C \bullet [h_{t-1}, x_t] + b_C\right) \tag{5.5}$$

$$C_u = i_t \otimes C_i \tag{5.6}$$

$$C_t = f_{go} \oplus C_u \tag{5.7}$$

Where σ symbolises sigmoid layer, $\otimes$ is pointwise multiplication, $\oplus$ is pointwise addition, W_i and W_C are weights, and b_i and b_C are biases.

(iii) **Output gate**

It is composed of sigmoid and a tanh layers which together determine the information output by the LSTM cell. It has 3 inputs: C_t (equation 5.7), x_t and h_{t-1}. Sigmoid

layer selects the part of current LSTM cell state to be included in the output (hidden state) as shown in equation 5.8. Tanh layer is used to shift the output in the range of $[-1, 1]$. Pointwise multiplication of O_t and C_t (computed using equation 5.9) is carried out to obtain the output (hidden state) of LSTM cell (h_t).

$$O_t = \sigma\left(W_o \bullet [h_{t-1}, x_t] + b_o\right) \tag{5.8}$$

$$h_t = O_t \otimes tanh\ \ (C_t) \tag{5.9}$$

Where σ symbolises sigmoid layer, $\otimes$ is pointwise multiplication, W_o are weights, and b_o is bias. It should be noted that tanh function is used because its 2^{nd} derivative is able to sustain for a long range before going to zero. This property helps in overcoming vanishing gradient problem. Sigmoid function is used because its output is in range $0 - 1$, that can be used to remember or forget the information.

5.4.4 Training of LSTM

Each vector output of embedding layer for each sub-word and input word are used to train individual LSTM networks. That is, if input word is subdivided to n sub-words, the number of LSTM networks are $n + 1$, the additional one being that of full input word. The networks are labelled as $LSTM_i$, where i refers to index of LSTM network. In this work, $LSTM_0$, $LSTM_1$, and $LSTM_2$, respectively refers to LSTM networks for full input word, 1^{st} sub-word, and 2^{nd} sub-word.

For all networks, 3 layers were used. Numerical vectors representing input word and sub words are fed to respective LSTM network where data passes through various LSTM cells at various time steps. The outputs of last LSTM cells of all networks are then concatenated to form one combined vector (f_c) as shown by pseudo-code in equation 5.10. The combined vector is then input to dense layer for final classification. Tables 5.1 and 5.2 show hyperparameters used for LSTM and dense layer networks.

$$f_c = Concat\,(f_0, f_1, f_2) \tag{5.10}$$

Where f_0, f_1, and f_2 are respectively outputs of last LSTM cells of $LSTM_0$, $LSTM_1$, and $LSTM_2$ networks.

Table 5.1: Hyperparameters of LSTM networks

Layer	Number of units			Drop out
	$LSTM_0$	$LSTM_1$	$LSTM_2$	
1	256	128	128	0.4
2	256	128	128	0.4
3	256	128	128	0.4

Table 5.2: Hyperparameters for dense (fully connected) layers

Layer	No. of units	Activation function	Drop out
1	256	ReLU	0.5
2	128	ReLU	0.5
3(Output)	15	Softmax	-

During training with LSTM networks (figure 5.1), global features are learnt and extracted using $LSTM_0$ network. Local features are learnt and extracted using $LSTM_1$ and $LSTM_2$ networks. Global features refer to features pertaining to full input word while local features refer to features pertaining to sub-words. f_c being an aggregation of local and global features, it models well input words of different languages. For dense layers, the last layer (layer 3) is classification layer and its number of units correspond to number of languages in train set.

5.5 Experimental Results

5.5.1 Evaluation Datasets

The proposed Frag-LSTM-LID method has been evaluated with two datasets: Universal Declaration of Human Rights (UDHR) corpus [183, 184, 185] and Kenya indigenous languages

(KIL) corpus. UDHR dataset consists of declaration fundamental human rights textually translated to 360 different languages and dialects across the world, available in PDF format. On December 10^{th} 1948, United Nations General Assembly in Paris adopted the declaration as basic and fundamental human rights for all peoples and all nations. Of these, 281 translations have been converted to text since others are poor quality handwritten scans.

The **Kenyan indigenous languages (KIL)** corpus has been created by the author consisting of under-resourced languages from Kenya. It was prepared by collecting many text sentences from 15 indigenous languages spoken in Kenya. The languages are Kamba (kam), Kikuyu (kik), Kipsigis (kip), Kisii (kis), Luhya (luh), Luo, Maasai (maa), Meru (mer), Nandi (nan), Oromo (oro), Samburu (sam), Somali (som), Swahili (swa), Taita (tai), and Teso (tes). The languages were selected because they are the top indigenous languages commonly used by majority of Kenyan people beside official language, English. The texts were extracted from Bibles written in those languages. The bibles were used as source document because they are easily available. In addition, standard language structure is used while translating the Bibles to those indigenous languages. The dataset is then split into train and test sets for every language texts.

5.5.2 Evaluation metrics

Performance metrics used to evaluate Frag-LSTM-LID technique are F1-score, precision (P), accuracy, and recall (R). Confusion matrix is also used.

(a) **Accuracy**

This is the proportion of correct language predictions for a given number of predictions. It is computed using equation 5.11. In this work, accuracy is computed over entire dataset.

$$Accuracy = \frac{TN + TP}{(FN + FP + TN + TP)} \tag{5.11}$$

Where TP is true positives, TN is true negatives, FP is false positives, and FN is false negatives.

(b) **Precision (P)**

It is the proportion of correct predictions for a given total positive predictions. It is computed by equation 5.12.

$$P = \frac{TP}{(FP + TP)} \tag{5.12}$$

High values of precision is an indicator of low cases of false positives (FP), hence an efficient LID system.

(c) **Recall (R)**

Is the ration of correct predictions to actual positives as shown in equation 5.13.

$$R = \frac{TP}{(FN + TP)} \tag{5.13}$$

(d) **F1-Score**

Is weighted mean of recall (R) and precision (P) as given by equation 5.14.

$$F1 - Score = 2 \times \frac{P \times R}{P + R} \tag{5.14}$$

F1-score is a better and reliable measure of goodness of a given technique.

(e) **Confusion matrix**

This is a 2D $n \times n$ table or matrix that shows misclassification/misprediction (or classification misses) of a model. In the matrix, one axis (say x-axis) is actual class/label and another axis (say y) represents predicted class. The main diagonal (running from top left to bottom right) shows the correct predictions. The off main diagonal values (i.e., in upper and lower triangles) represents classification/prediction misses w.r.t actual/true class labels.

5.5.3 Performance of proposed Frag-LSTM-LID method

Table 5.3 shows performance of the Frag-LSTM-LID technique when evaluated with UDHR corpus [183, 184, 185] and Kenya indigenous languages (KIL) corpus. It can be seen from the table that Frag-LSTM-LID method attains high scores of 0.995, 0.995, 0.998, and 0.97 for precision, recall, Accuracy, and F1-measure metrics for KIL corpus of indigenous languages in Kenya.

For UDHR corpus [183, 184, 185], Frag-LSTM-LID technique attains 0.95, 0.96, 0.935, and 0.954 respectively for F1-measure, accuracy, recall, and precision metrics. The high scores shows that the technique has high efficiency and models well various languages. This is attributed to various pathways of LSTM networks ($LSTM_0$, $LSTM_1$, and $LSTM_2$ in figure 5.1) that extracts well global and local features that are then aggregated to form a combined feature vector f_c (section 5.4). These combined features have high discriminating powers, thus enabling to distinguish various languages.

Table 5.3: Performance of Frag-LSTM-LID technique in language identification with UDHR [183, 184, 185] and Kenya indigenous languages (KIL) datasets

Dataset	Precision	F1-Score	Accuracy	Recall
UDHR [184, 183, 185]	0.954	0.95	0.96	0.935
KIL	0.995	0.97	0.998	0.995

However, it can be seen from table 5.3 that scores for UDHR dataset [183, 184, 185] are slightly lower than those for KIL corpus. This is due to large number of languages (291) in UDHR dataset [183, 184, 185], which causes higher misclassification due to possible increase in number of related languages.

Table 5.4 shows confusion matrix of Frag-LSTM-LID model trained with KIL corpus. In the matrix, x-axis represents true/actual language label while y-axis represents the predicted language class. From the matrix's main diagonal, the large values show good performance of the proposed Frag-LSTM-LID technique. Main diagonal shows cases where number of predictions and true labels are equal, i.e., correct predictions. The low values in lower and upper triangles shows low misclassifications/confusion of the proposed method in LID. In addition, the confusion matrix shows languages that are related by their higher misclassification values w.r.t other languages. For example, Nandi (nan) and Kipsigis (kip) are related

languages since their misclassification value is 0.128. The two languages belong to the same language group.

Table 5.4: Confusion matrix for KIL corpus

kam	kik	kip	kis	luh	luo	maa	mer	nan	oro	sam	som	swa	tai	tes	
kam	0.994	0.004	0	0	0	0	0	0.002	0	0	0	0	0	0	0
kik	0.002	0.975	0 0.013	0	0	0	0	0	0	0	0	0	0	0	
kip	0	0	0.978	0	0	0	0	0.128	0	0	0	0	0	0	0
kis	0	0.001	0	0.995	0	0	0	0.002	0	0	0	0	0.002	0	0
luh	0	0	0	0.015	0.981	0	0	0.003	0	0	0	0	0	0	0
luo	0	0	0	0	0	0.997	0	0	0	0.002	0	0	0	0	0
maa	0	0	0	0	0	0	0.98	0	0	0.007	0.009	0.003	0	0	0
mer	0.002	0.013	0	0	0	0	0	0.984	0	0	0	0	0	0	0
nan	0	0.125	0	0	0	0	0	0	0.872	0	0.003	0	0	0	0
oro	0	0	0	0	0	0	0	0	0	0.999	0	0	0	0	0
sam	0	0	0	0	0	0	0	0	0	0.01	0.988	0	0	0	0
som	0	0	0	0	0	0	0	0	0	0	0	0.999	0	0	
swa	0	0	0	0.005	0	0	0	0	0	0	0	0	0.982	0.009	0.003
tai	0.001	0	0	0	0	0	0	0.007	0	0	0	0	0.008	0.983	0
tes	0	0	0	0.005	0.008	0	0	0.002	0	0	0	0	0	0	0.985

5.6 Conclusion

A word level language identification method called fragmented LSTM for language identification (Frag-LSTM-LID) has been proposed. In this method, an architecture consisting of 3 LSTM networks is used to extract local and global information from a word/string. The first network processes full word, thus extracting features of entire word (i.e., global features). A full word is subdivided to 2 equal or almost equal sub-words which are then processed by 2 separate LSTM networks to extract local features (features of sub-words). Global and local features are then aggregated by concatenation to single combined feature vector which is then used with dense layer for final classification. Frag-LSTM-LID method has been evaluated with UDHR and Kenyan indigenous languages (KIL) datasets. KIL corpus was developed by the author where it consists of texts of top 15 Kenyan indigenous languages extracted from Bibles written in those indigenous languages. Desirable performance was obtained for both datasets.

Chapter 6

BI-DIRECTIONAL FRAGMENT NETWORKS FOR DATING HISTORICAL MANUSCRIPTS

6.1 Introduction

Historical manuscript dating (HMD) is a task of determining era or creation period/time of historical manuscripts. Production time or age of historical manuscripts is an attribute of interest to concerned stakeholders like curators, archivists, historians, and paleographers. The age or creation time can in turn be used in classification, indexing, and determining monetary value of historical manuscripts. Production time of undated manuscripts can be determined by analysis of its layout, contents, material used, or from author if known. Age or production time is an implicit attribute that cannot be directly gleaned from a manuscript. It is not easy to obtain it compared to explicit attributes [186] like colour texture, layout, and handwriting style. Age, like other implicit attributes, is often inferred from explicit attributes like the ones mentioned before.

A computer-based era estimation method uses spatio-temporal information like handwriting style of historical manuscript images (HMI). Handwriting style is spatio-temporal in that it evolves gradually over time due to various factors [62] like age, culture, gender, education

level, time, emotions, writing speed, writing surface, and writing instrument. This evolving over time is in structure of some characters/letters as shown in Figure 6.1 [62]. In the figure (Figure 6.1), evolving of writing styles of letters a, d, g, and p over a period of 250 years is shown. Over time, the structure of writing of the letters changed. Therefore, a feature that captures changing trend of handwriting style over years is used to time-stamp HMI, thus estimating its era.

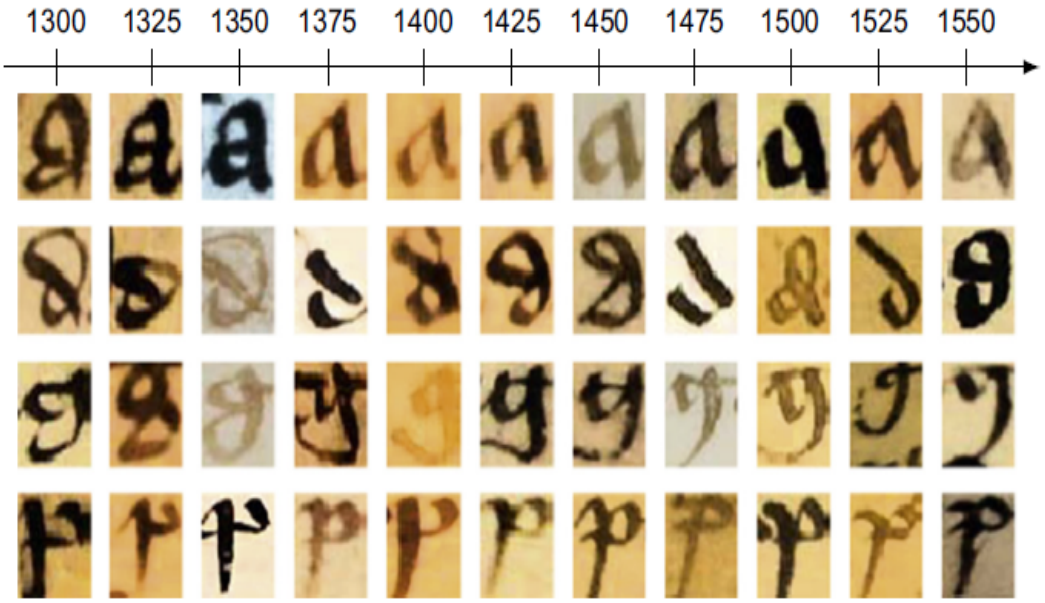


Figure 6.1: Evolving over time (horizontally) of handwriting styles for letters a, d, g, and p respectively from top downwards [62]

This chapter presents a deep learning-based technique called Bi-directional fragment network (BiD-FragNet) for dating historical manuscripts. The proposed method has been inspired by FragNet method by He and Schomaker [52] originally used for writer identification. BiD-FragNet is used to learn and extract temporal information in handwriting styles of different authors for dating task. BiD-FragNet consists of 2 CNN network pathways (main and fragment pathways) through which features are learnt and extracted. A manuscript is segmented into square patches that are fed into the 2 CNN pathways. The extracted features are used to build a dating model by training MLP classifier. The main pathway is used to learn and extract global information whereas fragment pathway learns and extracts local information. In the proposed technique, there is bi-directional sharing of information between main

and fragment pathways, hence the network referred to as bi-directional fragment network (BiD-FragNet). This bi-directional sharing of information makes the final features obtained capturing more information and hence having high discriminating power compared to that of original FragNet [52]. The method is explained in detail in section 6.3. The proposed BiD-FragNet differs from the original FragNet [52] in the way information is shared between main and fragment channels. In the proposed BiD-FragNet, there is bi-directional sharing of information between main and fragment channels whereas in original FragNet [52], sharing of information between main (feature pyramid) and fragment channels is unidirectional, i.e., information flows only from main (feature pyramid) channel to fragment channel.

6.2 Problem formulation

Most of HMD tasks are based on traditional methods which consist of physical and paleographical techniques [5]. These methods require actual manuscript sample, are slow, and impractical when manuscripts are very many. The need of actual sample by traditional methods makes them destructive. Physical techniques use costly and specialized equipments hence not affordable/available to many people. Paleographical methods are subjective since they rely on human experts making their results to be prone to errors. This is seen in cases where different paleographers arrive at different dates/ages for the same manuscript. To mitigate the destructive, efficiency, and speed issues associated with traditional HMD methods, computer-based HMD techniques are needed because they use digital images of historical manuscripts. There are many computer-based HMD methods in literature but their performances are not satisfactory. In light of this, a deep learning-based BiD-FragNet method is proposed to address the problem. The method uses 2 CNN pathways to predict production date or era of a historical manuscript images. The pathways share information at set intervals during feature extraction phase.

6.3 BiD-FragNet for historical manuscript dating

BiD-FragNet is composed of 2 pathways: main and fragment pathways. The pathways are joined together by crossed connections at every level as shown in figure 6.2. The connections between channels are for level-wise bi-directional sharing of information between the pathways. Main pathway learns discriminative features at global scale whereas fragment pathway learns discriminative features at local scale. Feature maps (G_i) of convolution layers in main pathway are fragmented and then concatenated with feature maps (D_i) of convolution layers in fragment pathway as explained in section 6.3.1. In the same way, feature maps (D_i) of convolution layers in fragment pathway are defragmented and then concatenated with feature maps (G_i) of convolution layers in main pathway as will be explained in section 6.3.1. The BiD-FragNet framework (figure 6.2) is explained in detail in sub-sections that follow.

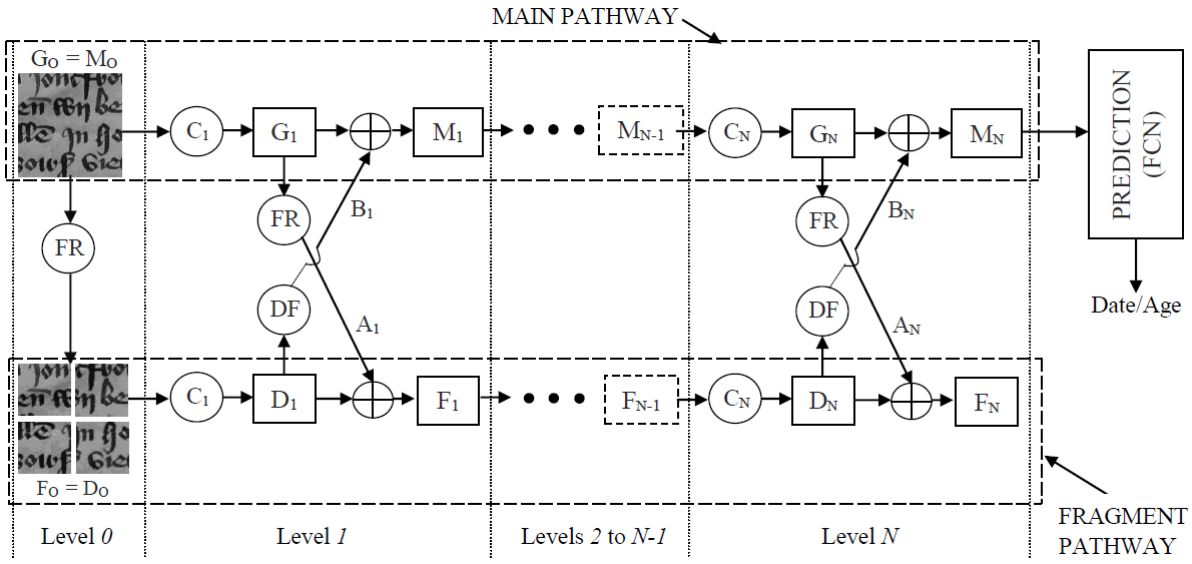


Figure 6.2: Framework of BiD-FragNet consisting of main and fragment pathways, where © denotes convolution for i^{th} level, G_i denotes feature maps of i^{th} level convolution layers in main pathway, D_i denotes feature maps of i^{th} level convolution layers in fragment pathway, FR denotes fragmentation of G_i to obtain A_i, DF denotes defragmentation of D_i to obtain B_i, M_i denotes result of i^{th} level concatenation of G_i and B_i, F_i denotes result of i^{th} level concatenation of D_i and A_i, and $\bigoplus$ denotes concatenation

6.3.1 Fragmentation (FR) and defragmentation (DF)

Fragmentation (FR) is the process of subdividing feature maps (G_i) of convolution layers in main pathway to obtain non-overlapping fragment feature maps (A_i) of smaller dimensions as shown in figures 6.2 and 6.3. This process is done at every level/scale, and such that dimensions of A_i is same as that of D_i (output of convolution layer in fragment pathway). A fragment feature map (A_i) is expressed by equation 6.1 [52].

$$A_i = Crop\left(G_i, \theta = (r, c, h, w)\right) \tag{6.1}$$

Where G_i is input feature map, $\theta = (r, c, h, w)$ are parameters of location of cropping and dimensions of resulting fragment (A_i). r and c are respectively rows and columns in G_i where segmentation starts, w is width and h is height of the resulting fragment map (A_i). In this work, $h = h_{Gi}/2$ and $w = w_{Gi}/2$ where h_{Gi} and w_{Gi} are respectively height and width of G_i. Consequently, $r = \{0, h_{Gi}/2\}$ and $c = \{0, w_{Gi}/2\}$ where h_{Gi} and w_{Gi} assume meanings as before. A_o denotes fragments obtained from input image G_o.

Defragmentation (DF) is the process of joining together of feature maps (D_i) of convolution layers in fragment pathway to obtain defragmented feature maps (B_i) as shown in figures 6.2 and 6.3. As shown in figures 6.3, a feature map G_i from main pathway (figure 6.3 left) is fragmented along the middle axis to produce 4 fragments and 4 features maps (D_i) from fragment pathway (figure 6.3 right) joined together to obtain one big feature map (B_i). Defragmentation is done by joining 2 feature maps D_i along each dimension (vertical and horizontal). The resulting defragmented feature maps (B_i) have same dimensions as G_i (output of convolution layer in main pathway) of corresponding level in main pathway. This process also is done at every level. A defragment feature map (B_i) is expressed by equation 6.2.

$$B_i = Concatenate\left(D_i^{h/2 \times w/2}, \theta = (h, w)\right) \tag{6.2}$$

Where $h/2$ and $w/2$ are respectively height and width of D_i, and h and w are respectively

height and width of resulting defragmented feature map (B_i).

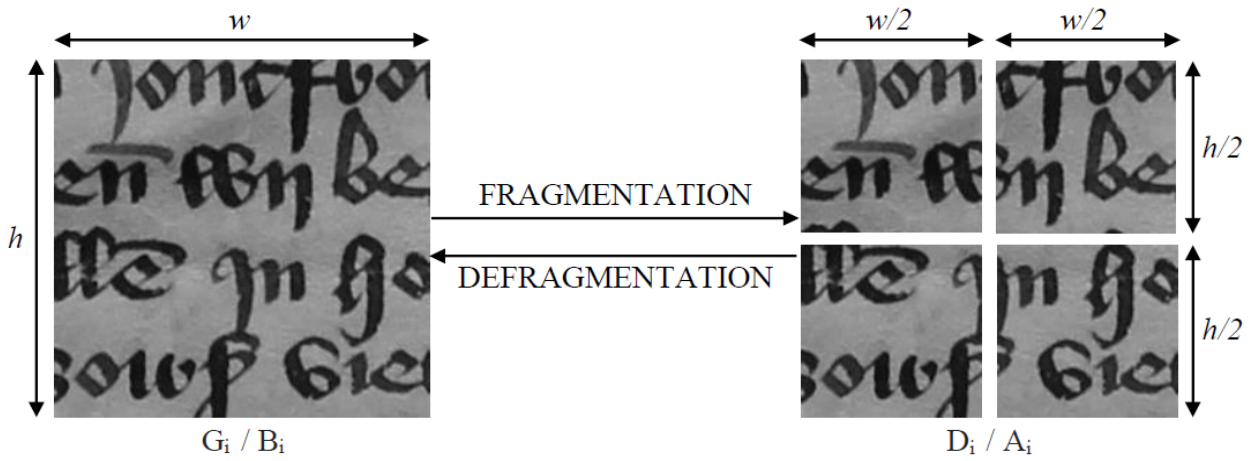


Figure 6.3: Fragmentation and defragmentation in i^{th} level in BiD-FragNet network where G_i is feature map of convolution layer in main pathway, B_i is defragmented feature map obtained by joining together feature maps D_i. D_i are feature maps of convolution layer in fragment pathway, and A_i are fragment feature maps resulting from fragmentation of G_i

6.3.2 Main pathway

It consists of traditional CNN [51, 187, 188] architecture. Inputs to main pathway are 256×256 gray images (patches) of scanned manuscript documents. At every level in main pathway, there are convolution, max pooling, and concatenation processes as will be explained shortly.

In BiD-FragNet system (shown in figure 6.2), feature maps (G_i) from convolution layer outputs of main pathway are concatenated with defragmented feature maps (B_i) from same level convolution layers in fragment channel to obtain main channel concatenated feature maps (M_i). B_i is obtained by defragmenting (joining together) feature maps (D_i) of same-level convolution layer outputs in fragment pathway as explained in section 6.3.1. Dimensions of B_i and G_i are same. The concatenated feature maps (M_i) are expressed as shown in equation 6.3.

$$M_i^{h \times w \times j_m} = \oplus (G_i, B_i) \quad \forall\ i > 0 \tag{6.3}$$

Where $\oplus$ denotes concatenation operation, G_i and B_i are input feature maps, $h \times w \times j_m$

are dimensions of resulting concatenated feature map (M_i) where w is width and h is height of M_i which is same as that of G_i and B_i. $j_m = c_{Gi} + c_{Bi}$ is number channels of M_i where c_{Gi} and c_{Bi} are respectively number of channels of G_i and B_i. M_i becomes input of next level ($(i+1)^{th}$ level) convolution layer in main pathway. M_o is a special case where it is input image for main pathway, and $M_0 = G_0$. Feature map (G_i) is mathematically expressed as in equation 6.4.

$$G_i = \bigotimes \left(M_{i-1}, W^{m \times n \times n \times k} \right) \qquad \forall\ i > 0 \tag{6.4}$$

Where $\bigotimes$ denotes convolution operation, M_{i-1} is concatenated feature map of previous level ($(i-1)^{th}$ level) in main channel. $W^{m \times n \times n \times k}$ is convolution kernel of shape $m \times n \times n \times k$ where m is number of channels of input feature maps, n is kernel size (3), and k is number of channels in output feature map. It should be noted that $m = c_f\,(i-1) + c_m\,(i-1)$ where $c_f\,(i-1)$ is number of channels of feature maps (D_i) of convolution layer in previous level ($(i-1)^{th}$ level) in fragment pathway, and $c_m\,(i-1)$ is number of channels of feature maps (G_i) of previous ($(i-1)^{th}$ level) convolution layer of main pathway.

The main pathway fuses information/features from 2 sources: (a) previous convolution layer in main pathway, and (b) feature maps defragmented from output of convolution layer in same level in fragment pathway. Max pooling with size 2×2 and step-size of 2 is performed after every 2 convolution layers (levels) so as to get: (i) translation invariance, (ii) more refined feature maps at every level, and (iii) feature maps of reduced size. In BiD-FragNet, 6 layers or levels are used in main pathway which is same in fragment pathway as explained in the next section.

6.3.3 Fragment pathway

Fragment pathway also consists of traditional CNN [51, 187, 188]. Input to fragment pathway are 64×64 gray images (sub-patches) obtained by fragmenting images input to BiD-FragNet system (of dimension 256×256). The input images (to main pathway) are fragmented as explained in section 6.3.1. Thus, for every 256×256 input image, four (4) 64×64 fragments

(patches) are obtained. These fragments are then input to fragment pathway. In fragment pathway, convolution, max pooling, and concatenation processes are carried out as will be explained later. Convolution output in fragment pathway (D_i) is mathematically expressed by equation 6.5.

$$D_i = \otimes \left(F_{i-1}, W^{m_f \times n \times n \times k_f}\right) \qquad \forall\ i > 0 \tag{6.5}$$

Where $\otimes$ denotes convolution operation, F_{i-1} is concatenated feature map of previous level ($(i-1)^{th}$ level) in fragment channel. $W^{m_f \times n \times n \times k_f}$ is convolution kernel of shape $m_f \times n \times n \times k_f$ where m_f is number of channels of input feature maps, n is kernel size (3) and k_f is number of channels in output feature map. As stated in section 6.3.1, $m_f = c_f\,(i-1) + c_m\,(i-1)$ where $c_f\,(i-1)$ is number of channels of feature maps (D_i) of convolution layer in previous level ($(i-1)^{th}$ level) in fragment pathway, and $c_m\,(i-1)$ is number of channels of feature maps (G_i) of previous ($(i-1)^{th}$ level) convolution layer of main pathway.

Feature maps (D_i) from convolution layer outputs of fragment pathway are also concatenated with fragmented feature maps (A_i) from same level convolution layers in main channel to obtain fragment channel concatenated feature maps (F_i). Feature maps F_i are mathematically expressed as shown in equation 6.6.

$$F_i^{h \times w \times j_f} = \oplus\,(D_i, A_i) \quad \forall\ i > 0 \tag{6.6}$$

Where $\oplus$ denotes concatenation operation, $h \times w \times j_f$ are dimensions of resulting concatenated feature map (F_i) where $h \times w$ are height and width dimensions of F_i which is same as that of D_i and A_i. D_i and A_i are input feature maps, and $j_f = c_{Di} + c_{Ai}$ is number channels of F_i where c_{Di} and c_{Ai} are respectively number of channels of D_i and A_i. F_i becomes input of next level ($(i+1)^{th}$ level) convolution layer in fragment pathway. F_o is a special case where it is input image for fragment pathway, and $F_0 = D_0$.

As in main pathways, max pooling with size 2×2 and step-size of 2 is performed after every 2 convolution layers (levels) for the purposes of achieving translation invariance, more refined feature maps at every level, getting feature maps of reduced size. 6 layers or levels are used

in fragment pathway where each level consists of convolution, max pooling, concatenation of features maps, and bi-directional sharing of information between the two pathways. Also, fragmentation and defragmentation of feature maps are carried out as afore-explained.

In BiD-FragNet system, level-wise sharing of information causes refined, efficient, and features with high discriminating powers to be learned at both local and global scales. The main pathway learns features in global scale whereas fragment pathway mainly extracts local feature of text-blocks of manuscripts [55, 189]. Error between target and network prediction is computed using cross entropy loss function [190] and Adaptive moment (Adam) estimation [123] is used for optimization process.

Concatenated feature maps (M_i) obtained from last level in main pathway are input to fully connected network (multi-layer perceptron) for classification.

6.4 Results and Discussion

In this section, performance of BiD-FragNet method for manuscript dating is discussed. Performance of BiD-FragNet is also compared with state-of-the-art HMD techniques. Medieval Paleographical Scale (MPS) dataset [77] of historical manuscripts was used to evaluate the proposed BiD-FragNet for HMD. The dataset consists of 2858 scanned images of Charters collected from 4 cities. The dataset is arranged into 11 key-periods (classes) from 1300-1550 CE, where each key-period spans 25 years. Each key period (class) contains manuscripts written within a period of 25 years. Professionals wrote the original manuscripts. The MPS dataset [77] consists of scanned historical manuscripts that depict gradual evolving of writing styles over time.

The MPS dataset [77] was first randomly split to train (70%), test (15%), and train (15%) sets at key-period level. Each manuscript image in all sets (train, validation, and test) are randomly split to 256×256 patches, which are converted to gray scale and then input to BiD-FragNet dating system. Features are learnt and extracted from the manuscript patches via main and fragment pathways as explained in section 6.3. Since the inputs to BiD-FragNet are manuscript patches, the outputs are predicted creation years (classification scheme) of

the patches. Creation year (or date) of a given manuscript document is the year predicted (by BiD-FragNet) for majority of patches cropped from that manuscript document.

6.4.1 Evaluation metrics for manuscript dating

Performance metrics for evaluating BiD-FragNet method for manuscript dating are cumulative score (CS) and mean absolute error (MAE).

(a) **Mean absolute error (MAE)**. This is the mean of absolute differences between actual and estimated era or creation/production times of historical manuscripts as expressed in equation 6.7 [75, 191].

$$MAE = \frac{1}{N}\sum_{i=1}^{N}|y_i - t_i| \tag{6.7}$$

Where $i = 1, 2, 3, ..., N$ is index of a manuscript, N is total number of manuscripts in a test dataset, and t_i and y_i are actual and estimated eras/production times respectively of i^{th} manuscript. Small MAE values show that the dating model is efficient.

(b) **Cumulative Score (CS)**. It is given by equation 6.8 [78, 192].

$$CS\,(\alpha) = \frac{N_{e\leq\alpha}}{N} \times 100 \tag{6.8}$$

Where $N_{e\leq\alpha}$ is the number of manuscripts whose estimated production time/era makes absolute error (e) not higher than α years (or any other units of production times and N is total number of historical manuscripts in a test set. Higher CS value show that the dating model is better and efficient.

6.4.2 Performance of various techniques with MPS dataset [77]

Table 6.1 shows comparison of performances of the proposed BiD-FragNet for HMD and state-of-the-art methods for MPS data set [77]. The evaluation scores of other methods are

obtained from work by Hamid et al [78]. From the table, it can be seen that BiD-FragNet method has the best performance with MAE of 13.45 years.

From the Table 6.1, the techniques using PSD [62] and CF-deg3 [75] comes 2^{nd} and 3^{rd} respectively whereas the method using Hinge [44] has lowest performance. This good performance of the proposed BiD-FragNet can be attributed to efficient handwriting style features with high discriminating features that are learnt and extracted by the proposed BiD-FragNet. The features are learnt at local and global scales by main and fragment pathways respectively as explained in section 6.3. These local and global information is refined and shared between main and fragment pathways of BiD-FragNet at set levels, hence efficiently capturing inherent temporal information from handwriting styles in historical manuscripts.

Table 6.1: Comparative performance evaluation of BiD-FragNet dating method for MPS data set [77]

Feature/method used	MAE	CS($\alpha = 25$)
Hinge[44]	30.3	68.5
Quill[45]	45.1	60.0
Junclets [71]	25.6	73.6
Gabor+LBP+U-LBP[78]	20.13	-
PSD [62]	15.1	87.4
SF [75]	26.0	73.2
CF [75]	17.9	81.0
BiD-FragNet	**13.45**	**94.0**

CF: contour fragments
SF: stroke fragments
PSD: polar stroke descriptor
LBP: local binary pattern

Figure 6.4 shows cumulative score probability for various error (α) values from $0-200$ years for MPS data set [77]. It is seen that the BiD-FragNet method has better performance for $\alpha > 50$ where $CS > 94\%$. This is consistent with the fact that handwriting style change remarkably after 50 years, which can be due to evolving culture, age, and educational level/training. The least performing methods are those using Delta-Hinge [193] and Quill-Hinge [45] features. They perform dismally for $\alpha < 50$ where $CS < 85\%$ as compared to the

proposed method whose $CS < 94\%$over the same error value.

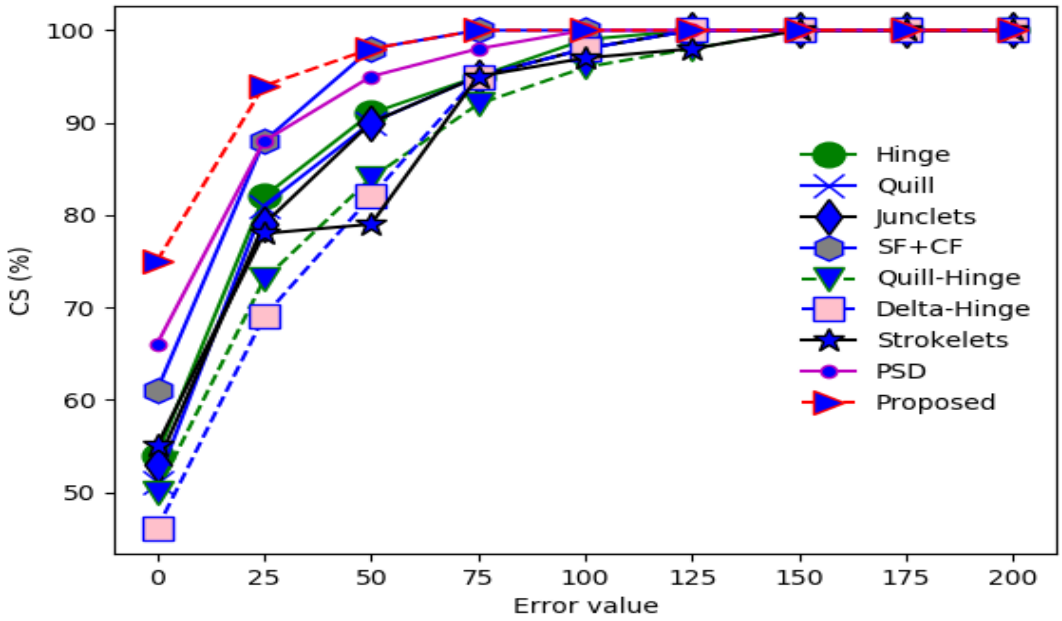


Figure 6.4: Cumulative score for hinge [44], quill [45], junclets [71], $SF+CF$ [75], quill-hinge [45], delta-hinge [193], strokelets [76], polar stroke descriptor (PSD) [75] features, and the proposed BiD-FragNet for various error values for MPS data set [77]

Performance of BiD-FragNet method is also evaluated key year-wise using confusion matrix of predicted year of creation as shown in table 6.2. The confusion matrix shows probability of manuscript in key year t_i that is estimated by the proposed BiD-FragNet method to belong to key year y_j, where t_i and y_j are respectively ground truth and predicted creation year/time, and i & j are in same range. From the confusion matrix (table 6.2), it can be seen that diagonal values (where $i = j$) are high showing good performance of BiD-FragNet for dating historical manuscripts. Diagonal values are those of correct prediction of manuscript's production year. It can also be seen that predicted probabilities are high for key years (periods) near ground truth key years as compared to those far. This is consistent with the evolving of handwriting style with time. There is low between variability for key years (periods) within a small range where it is expected that handwriting has not evolved so much whereas there is relatively large between variability for key years (periods) within a wide range where handwriting has evolved appreciably. From the confusion matrix in table 6.2, the values off the main diagonal are low showing that the proposed BiD-FragNet method has low prediction misses, further showing its efficiency in dating historical manuscript images

as already mentioned before.

Table 6.2: Confusion matrix for performance evaluation of the proposed dating method for MPS data set[77]

		PREDICTED CREATION YEAR										
		1300	1325	1350	1375	1400	1425	1450	1475	1500	1525	1550
GROUND TRUTH	1300	47	27	14	6	3	0	2	1	0	0	0
	1325	20	57	18	2	2	1	0	0	0	0	0
	1350	12	21	35	18	10	2	2	0	0	0	0
	1375	2	9	19	29	25	11	5	0	0	0	0
	1400	0	0	2	14	45	23	14	0	2	0	0
	1425	0	0	1	9	20	36	23	11	0	0	0
	1450	0	0	0	3	14	25	27	23	8	0	0
	1475	0	0	1	3	0	8	22	38	23	5	0
	1500	0	0	1	2	2	3	23	32	20	17	0
	1525	0	0	1	2	0	3	10	21	20	25	18
	1550	0	0	0	0	0	2	5	20	29	34	10

6.5 Conclusion

A deep neural network (DNN)-based Bi-directional FragNet (BiD-FragNet) technique has been proposed for historical manuscript dating. BiD-FragNet consists of CNN-based main and fragment channels that respectively learn global and local features from texts from historical manuscript document images. The two channels share information one to another at every level, thus enabling learning of efficient features with high discriminating powers. The learnt features are extracted and used with MLP for final classification. The proposed BiD-FragNet has been evaluated with MPS dataset [77] of historical manuscripts and attained mean absolute error of 13.45 and cumulative score of 94% for error rate of 25 years. Future work will focus on investigating effect of using multiple sub-channels (i.e., fragment channels) on performance of BiD-FragNet in historical manuscript dating.

Chapter 7

FUNNELLING ENSEMBLE METHOD FOR WRITER IDENTIFICATION (FEM-WI) FOR HISTORICAL MANUSCRIPTS

7.1 Introduction

This chapter presents an offline model-based writer identification method for handwritten historical manuscripts. The technique is called funnelling ensemble method for writer identification (FEM-WI) which is text independent. The proposed method is a 2-level system of classifier ensembles for efficient writer identification. Most of the available historical manuscripts have unknown authorship. Thus, identification of their authorship is a necessary task. Writer identification is one of the main manuscript management tasks.

The reasons for writer identification include: (i) it facilitates classification, archiving, and indexing tasks, (ii) it helps to estimate age or production date of manuscripts, (iii) it helps searching of other related manuscripts, (iv) it helps to obtain worth of a manuscript, and (v) helps to understand manuscripts and the author.

7.2 Handwritten writer identification (HWI)

Handwritten writer identification (HWI) is a pattern recognition task of establishing author of a handwritten text based on its features. Normally, the text's author is identified from a pool of authors whose authored sample texts are available. Authorship is determined by comparing features of test text and those of labelled texts (i.e., texts with known authorship). The author of the labelled text that is most similar to test text is deemed to be the author of the test text. HWI is a case of one-to-one classification in that a handwritten text (HWT) is associated with only one author from a pool of labelled texts. In this classification scheme, output consists of classes of individual authors. HWI is applied in various areas like criminal justice systems [194] to establish author of threat letters and in banks for signature verification.

HWI methods can be divided into 2 main classes: handcrafted features-based (HFB) and deep learning-based (DLB) techniques. HFB writer identification techniques employ features manually extracted from text documents using specially dedicated algorithms [41, 42, 44, 195, 211]. DLB writer identification techniques employ features auto-learned from input text documents using deep neural networks (DNN) like convolutional neural network (CNN) [2, 51]. Writer identification methods using auto-learned features include those of He and Schomaker [186] and He and Schomaker [52]. An exhaustive survey covering features for writer identification can be found in the work by Dargan and Kumar [196].

In most HWI tasks, the methods used are model-based where extracted features are used to build HWI model by training suitable classifiers using machine learning methods. Most of the HFB and DLB techniques fall in this category. The HWI model basically models handwriting styles of a range of authors such that when features of a test handwritten text are presented at the input, the HWI system gives the author as the output. More on model-based writer identification can be found in survey on machine learning based writer identification by Rehman et al [197].

For writer identification tasks that do not employ models, nearest neighbour search approach is used. This approach uses computed distance between query word image and candidate word images in a database with known authorship. Author of a candidate word with least

distance is the author of the query word. Methods employing this approach include those of Bulacu and Schomaker [44] and Brink et al [45]. Differences in handwriting styles among authors make their identification (based on their handwritten texts) a possible and doable task. Factors that determine one's handwriting style include age, health, writing speed, time, writing surface, emotions, education level, location, and writing tool [198]. As pointed out by He and Schomaker [52], handwriting style features can be extracted from segmented words, text block, or whole page. Word-based HWI is most challenging because single words have limited handwriting style information as compared to text block or whole text page.

The funnelling ensemble method for writer identification (FEM-WI) technique used here has been inspired by the work of Esuli et al [199] where a similar method was used for cross-lingual-multi-label text classification. FEM-WI consists of 2 levels of trained models: level-1 and level-2 models. Level-1 writer identification models are developed by training different classifiers using different features. The classifiers can be of same or different types. The outputs of level-1 feature-specific models/classifiers consist of class probabilities. The union of outputs of level-1 classifiers are in turn used to train a single level-2 meta classifier to obtain meta-model which gives the final output. FEM-WI works by leveraging on different features for same word image and different feature-specific base classifiers as explained in sections 7.4 and 7.5. In 2^{nd} level classifier, HWTs are represented in a common feature space of posterior probabilities computed by 1^{st} level feature-specific base classifiers. In FEM-WI system, base classifiers are used to map different feature spaces to a common feature space. That is, a word image is represented in different feature spaces in level-1, which are then funnelled to a common feature space in level-2. With this, all feature spaces (irrespective of the kind and number) are used to train meta-classifier. Thus, writer identification of a query word using any one feature benefits from all other features used to train the meta-classifier, hence giving improved performance. In this way, writer identification of all HWTs benefits from information present in all other individual features, which are aggregated by meta-classifier. Funnelling is a case of heterogenous transfer learning in that transfer learning is carried out across domains of different feature spaces. As similarly explained in Esuli et al [199]) with regards to funnelling approach, an enhanced model is built by leveraging on all available train examples and their respective different features extracted.

7.3 Problem statement

With most of historical manuscripts having been digitized to digital images, automated methods for their author identification are necessary. This is because manual and human-based writer identification is time consuming, slow, tiresome, and not feasible in large scale.

7.4 Funnelling ensemble method for writer identification (FEM-WI)

The proposed FEM-WI consists of 4 main steps: (i) Raw feature extraction, (ii) Base classifier (h_n^1) training, (iii) Meta-feature computation, and (iv) Meta-classifier training. Framework of FEM-WI system showing the main steps is shown in figure 7.1. The main steps in FEM-WI system are discussed in sub-sections that follow.

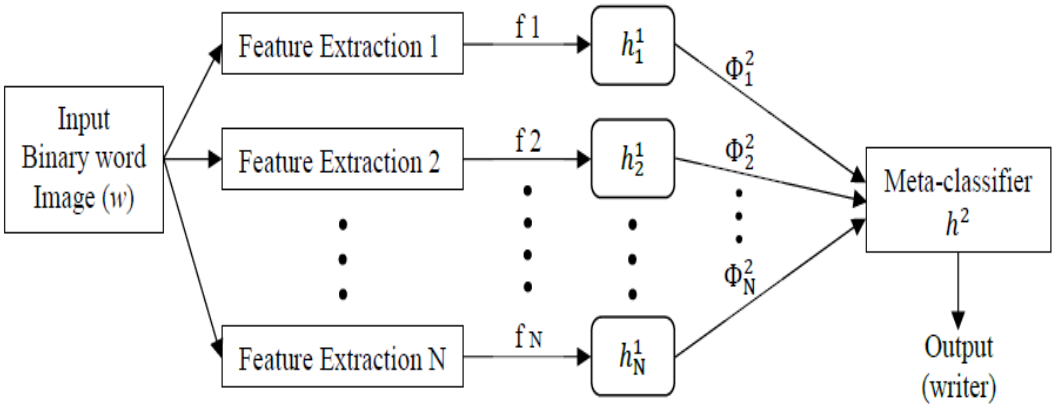


Figure 7.1: Framework of FEM-WI system where f_n^1 is n^{th} input/base feature descriptor, h_n^1 is n^{th} base classifier, and f_n^2 is n^{th} meta-feature descriptor obtained as output of h_n^1 and $n = 1, 2, ...N$ is the index of base/meta feature/classifier with N being number of features used, and h^2 is meta classifier.

The input to FEM-WI system are segmented word images (w). If the available data consists of scans of paragraphs or full-page documents, they are segmented to words using appropriate segmentation technique [200, 201] prior to input to writer identification system. Let $Tr = \{Tr_1, Tr_2, \ldots, Tr_L\}$ be available train set of handwritten segmented words of L classes, and $A = \{a_1, a_2 \ldots\ldots, a_L\}$ be the corresponding set of L authors. Let the train data (Tr) be

represented as $Tr/A = \{Tr_1/a_1, Tr_2/a_2, ..., Tr_L/a_L\}$ where Tr_i/a_i denotes that train class $Tr_i \in Tr$ consists of texts written by author $a_i \in A$.

At word level, i^{th} train class (Tr_i) is represented as $Tr_i = \left\{w_j^i\right\} = \cup_{j=1}^{M} w_j^i$ where $j = 1, 2, ..., M$ is word index and $i = 1, 2, ..., L$ is author index, with L being number of authors (classes) while M is number of words in a given class (written by a given author). M can vary depending on number of individual words written by respective authors $a_i \in A$. The inputs to FEM-WI system are segmented words represented as $w_j^i \in Tr_i$, where w_j^i denotes j^{th} handwritten word written by i_{th} author (a_i).

7.4.1 Raw feature extraction

Raw handcrafted features (f_n^1) are manually extracted from segmented words (w_j^i) using designed algorithms. Input word images are first binarized using MLP-based binarization method (discussed in chapter 3) such that background and foreground regions are respectively represented by 0 and 1. Contours of connected components (CC) of the binarized word images are extracted using a method by Schomaker and Bulacu [202]. The features are then extracted from the binarized word images and their contours. The features extracted are: histogram of orientation (HOO) pdf, fraglet histogram of oriented displacement (FHOD), relative run length (RRL), zoned relative run length (ZRRL), and modified contour hinge (mod-CH) pdf. Contour hinge PDF by Bulacu and Schomaker [44] is also extracted. The features are explained in detail in section 7.5.

7.4.2 Base classifier training

Feature-specific base classifiers (h_n^1) are trained with extracted raw features (f_n^1) (section 7.5) where $n = 1, 2, ..., N$ is the index of feature descriptor with N being number of features used. In other words, every individual base feature (f_n^1) is used to train separate base classifier (h_n^1). These base classifiers are referred to as tier-1 or level-1 classifiers. The superscript 1 in h_n^1 indicates that it is a tier-1 or level-1 classifier and subscript n denotes the index of feature descriptor used to train the classifier. Output of h_n^1 is a vector of dimension equal to

the number of output classes (L) (writers) of words in train data set. Thus, output vector's probability values correspond to L classes (writers), where each probability value shows the attribution confidence of h_n^1 that a word image (w_u) presented at its input is written by the corresponding author (classification scheme).

7.4.3 Meta-feature computation

Meta-features are computed as vector outputs of h_n^1 for various word image inputs. Met-features (f_n^2) are computed using k-fold cross validation (KFCV) approach. KFCV is a leave-one-fold-out approach, where train data is divided into k folds of equal size. Data division is done at class level so that every train class is equally represented in each fold. $k-1$ folds are used to train a classifier, and the one left out is used for testing and computing classifier vector outputs which are the meta features. The same is repeated each time a different fold left out till all folds are used to compute meta-features. This ensures that the data/fold used to compute meta-features has not been seen by the trained base classifier. KFCV approach is discussed in detail in section 7.7. The meta-features obtained have dimension equal to number of classes, and are used for training meta classifier (h^2).

7.4.4 Meta-classifier training

The meta-classifier (h^2) is trained using meta-features (f_n^2) which are outputs of base classifiers (h_n^1) for different raw feature descriptors (f_n^1), as afore-explained. Output of the meta-classifier gives the final identification of author for a given test word image.

7.5 Proposed features for writer identification

The proposed features for characterizing unique handwriting styles of various writers are: (i) Histogram of orientation (HOO), (ii) Fraglet histogram of oriented displacement (FHOD), (iii) Relative run length (RRL), (iv) Zoned Relative run length (ZRRL), and (v) modified contour hinge (mod-CH). HOO, FHOD, and mod-CH features are computed from contour

fraglets whereas RRL and ZRRL features are computed from white runs of binary handwritten word images with foreground and background regions respectively represented as 1 and 0. Contour fraglet here refers to a short section of a contour of a word consisting of consecutive contour pixels which are connected by 8-Neighbourhood connectedness. In this work, a fraglet is made up of 6 such contour pixels. A feature proposed by Bulacu and Schomaker [44] called contour hinge (CH) PDF is also used in this paper. In computing the proposed features, word image is first binarized using MLP-based binarization method (discussed in chapter 3) such that background and foreground regions are respectively represented by 0 and 1. Contours of connected components are then obtained from the binarized word images using a method by Schomaker and Bulacu [202]. The features used in this work are then computed from contours and the binary image obtained.

7.5.1 Histogram of orientation (HOO) pdf (*f1*)

Consider a fraglet AB bounded by a bounding box with center (P_c) as shown in figure 7.2, where P_i is i^{th} fraglet pixel. HOO is a feature that captures distribution of orientations (θ) of contour fraglet points with respect to the center (P_c) of the fraglet's bounding box. Orientation of fraglet pixel P_i is the computed angle (θ) between line P_iP_c and right-going horizontal axis starting at P_c as shown in figure 7.2. θ is in the range $0 - 360^o$.

Entire contour of a binarized word image is traversed, and at every contour pixel, a fraglet consisting of 6 consecutive pixels is obtained, and then orientations (θ) of its pixel points with respect to center (P_c) of fraglet's bounding box computed. The instance counts of orientations are accumulated into a histogram. The histogram is constructed such that its bins span 360^o. Bin size of 15^o is used hence 24 (360/15) bins are used.

7.5.2 Fraglet histogram of oriented displacement (FHOD) (*f2*)

This feature captures positional distribution of contour pixel points within fraglet region (fraglet's bounding box region). FHOD is computed with HOO as its basis as follows: for a fraglet AB (figure 7.2), obtain center P_c of its bounding box. For a fraglet pixel point P_i,

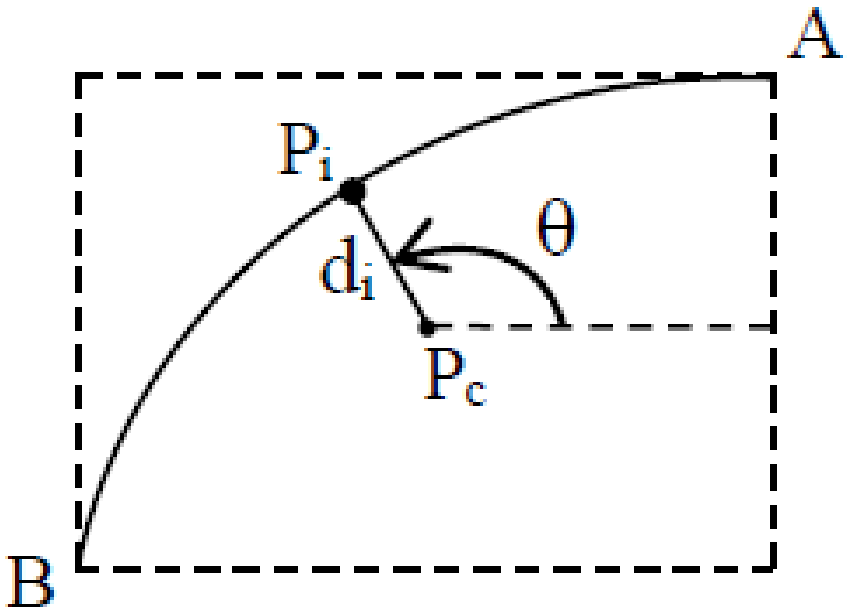


Figure 7.2: Computing histogram of orientation (HOO) and Fraglet histogram of oriented displacement (FHOD) features for fraglet AB, where P_i is fraglet pixel point, P_c is center of fraglet's bounding box, θ is orientation angle of P_i w.r.t P_c, and d_i is displacement of P_i from P_c

compute displacement (d_i) from P_i to P_c. Obtain orientation θ of P_i with respect to P_c as explained in section 7.5.1. θ is in the range $0 - 360^o$. The orientation range is split into 24 bins each of size of 15^o. The displacement d_i is voted/added to its respective angle bin. The same is repeated for all fraglet pixel points. Entire contour(s) of a word image is traversed, and at every contour point, a fraglet is obtained and the same process repeated to obtain oriented displacement of fraglet pixel points. The histogram obtained is then normalized to obtain FHOD.

7.5.3 Relative run length (RRL) (*f3*)

The use of run length for writer identification was pioneered by Arazi [203]. Run length captures distribution of white runs in a binary word image. In this work, white run (WR) refers to a group of connected pixels corresponding to continuous background region along a given axis (column or row) as shown for the word "god" in figure 7.3 as L_i, M_i, and R_i. Run length is the length (number of pixels) of a white run in a binary word image. White runs are

here grouped into 4 categories: left (L), middle (M), right (R), and empty (E) white runs. For horizontal/row runs, left white runs (L) are those enclosed by left vertical boundary of bounding box of a word and foreground pixel to the right as shown in figure 7.3a as L1, and L2. For vertical/column runs, L is one enclosed by top horizontal boundary of bounding box of a binary word image and foreground pixel to the bottom. Figure 7.3b shows L white runs for vertical white runs as L3 and L4. For horizontal runs, right (R) white run is one enclosed by right vertical boundary of bounding box of a word and a foreground pixel to the left as shown in figure 7.3a as R1 and R2. For vertical/column runs, R white run is one enclosed by foreground pixel at the top and bottom horizontal boundary of bounding box of a binary word image as shown in figure 7.3b as R3, and R4. Middle (M) white run is one that borders foreground pixels both at the left and right for horizontal runs as shown in figure 7.3a as M1, M2, M3, M4, and M5. For vertical runs, M white run is one that borders foreground pixels both at the top and bottom as shown in figure 7.3b as M6, M7, and M8. Empty (E) white run is one that is not bordered with a foreground pixel at its both ends. It runs from one bounding box boundary to another for a given axis/direction. Figure 7.3b shows E white run as E. A row/column can have only 1 or no L white run. The same applies for R white runs. However, a row/column can have 0 or multiple M white runs.

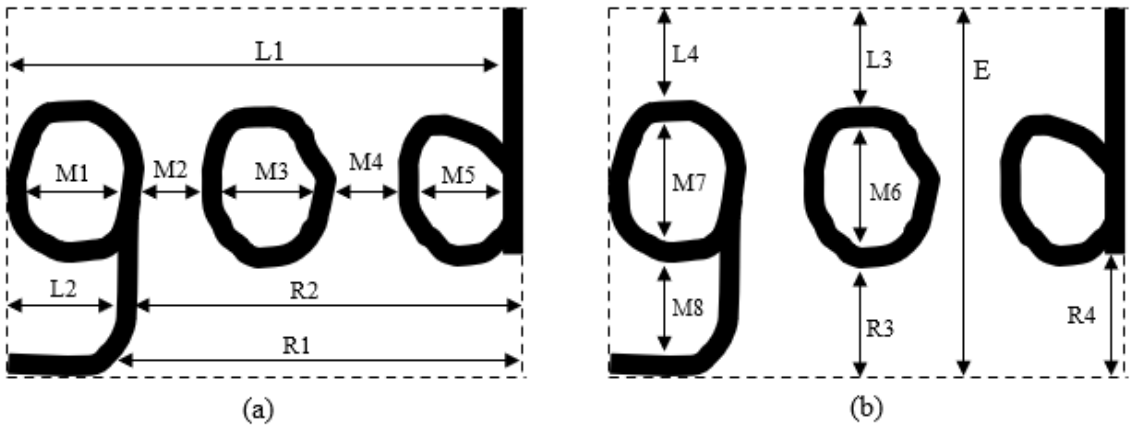


Figure 7.3: Categories of white runs for a binary word image (god) with foreground represented by 0 (black) and background region represented by 1 (white) where (a) shows horizontal white runs for 4 selected rows, and (b) shows vertical white runs for 4 selected columns. Dashed lines represent bounding box boundary lines

In computing RRL, the only WRs considered are L, M, and R. E white runs are not con-

sidered because they do not give any valuable information characteristic of a word or handwriting writing style of an author. RRL is computed as the length of WR expressed as a fraction of sum of white runs (of any category) in its immediate (adjacent) neighbourhood in same axis (column or row). For a row with N white runs: $WR_1, WR_2, ..., WR_N$, and their corresponding run lengths (RL), $RL_1, RL_2, ..., RL_N$, relative run length (RRL) of i^{th} WR is computed using equation 7.1. WR_i can be of same or different kinds. Distribution of RRL for word images captures characteristic handwriting style of different authors.

$$RRL_i = \frac{RL_i}{RL_{i-1} + RL_i + RL_{i+1}} \tag{7.1}$$

It should be noted that for a given row/column with multiple WRs, RRL for the 1st WR is its RL expressed as fraction of sum of RLs of the first 2 WRs. RRL for the last WR is its RL expressed as fraction of sum of RLs of the last 2 WRs. In cases where there is only 1 WR in a given row/column (like L1, R1, and R4 in figures 7.3(a & b) respectively), RRL is obtained as its RL expressed as a fraction of length of the corresponding axis. RRL ranges from $0 - 1$. For a $H \times W$ word image (god) shown in figure 7.3, expressions of RRL for various WRs is shown in table 7.1.

For all horizontal runs, RRLs computed for L are quantized into n bins and count for each instance represented as mini-histogram (L_{hist}) of n bins. Size of each bin is given as $1/n$. Bin index for each RRL is the integer value obtained from RRL/bin_size. The same is repeated for M and R white runs to obtain respective mini-histograms, M_{hist} and R_{hist}. The mini-histograms are then concatenated to obtain final relative run length (RRL_{hor}) feature vector as shown in equation 7.2.

$$RRL_{hor} = Concatenate\,(L_{hist}, M_{hist}, R_{hist}) \tag{7.2}$$

RRL_{hor} is then normalized to $0 - 1$ range. The same is repeated along the vertical (column) axis so as to obtain RRL_{vert} feature vector. The dimension of both RRL_{hor} and RRL_{vert} is 3n, where $n = 4$ is the number of bins used, thus dimensions of RRL_{hor} and RRL_{vert} are 12 in each. Both RRL_{hor} and RRL_{vert} are size invariant compared to ones used by Bulacu and

Table 7.1: Expressions for RRL for various WRs from $H \times W$ word image in figure 7.3

WR	RRL
L1	L1/W
L2	L2/(L2+R2)
L3	L3/(L3+M6)
M1	M1/(M1+M2)
M2	M2/(M1+M2+M3)
M3	M3/(M2+M3+M4)
M5	M5/(M4+M5)
M6	M6/(L3+M6+R3)
M7	M7/(L4+M7+M8)
R1	R1/W
R2	R2/(R2+L2)
R3	R3/(M6+R3)
R4	R4/H

Schomaker [44].

7.5.4 Zoned relative run length (ZRRL) (*f4*)

ZRRL uses RRL as the basis, with added information of white run's zone. Zoned relative run length (ZRRL) is computed as joint distribution of RRL and zone (z) of all WRs in a given axis of a binary word image. Zone (z) gives the position of a WR in the other axis. That is for instance, zone of horizontal WR (HWR) is its position on vertical axis and the zone of vertical WR (VWR) is its position on horizontal axis. Thus, zone is computed from axis other than axis of the given WR. The axis from which zone is computed is here referred to as zone axis, which is different from (and orthogonal to) axis of a given WR. Zone axis is divided to K zones of equal sizes, and the zone associated with a given WR determined as follows: Let $H \times W$ be dimensions of a binary word image where H and W are respectively height (number of rows) and width (number of columns). Let $i = 0, 1, 2, ..., H - 1$ be row index, and $j = 0, 1, 2, ..., W - 1$ be column index. The zone (z_k) of HWR on i^{th} row (i.e., HWR_i) is the integer k obtained from iK/H where K is number of zones used. For VWR on

j^{th} column (i.e., VWR_j), its zone (z_k) is also the integer k obtained from jK/W. Therefore, $k = 1, 2, ..., K$ is index of the zone (i.e., position of WR in zone axis). Same number of zones are used in both horizontal and vertical axes.

For horizontal WRs, count of joint instances of RRL_n and respective z_k for each of L, M, and R white runs (section 7.5.3) are recorded in 2D array of bins where $n = 1, 2, ..., N$ is bin index and $k = 1, 2, ..., K$ is zone index associated with RRL for HWRs. This 2D array is the mini-histogram of horizontal ZRRL (h-ZRRL). The mini-histograms of h-ZRRL for L, M, and R white runs are respectively represented as L_{h-ZRRL}, M_{h-ZRRL}, and R_{h-ZRRL}. Each of the h-ZRRL mini-histograms has dimension of NK. The mini-histograms are concatenated to form final horizontal zoned-relative run length ($ZRRL_{hor}$) feature vector as shown in equation 7.3.

$$ZRRL_{hor} = Concatenate\left(L_{h-ZRRL}, M_{h-ZRRL}, R_{h-ZRRL}\right) \tag{7.3}$$

The final feature vector $ZRRL_{hor}$ is normalized to a probability density function (pdf). The same process is repeated along the vertical axis to obtain vertical ZRRL feature vector ($ZRRL_{vert}$). The dimension of $ZRRL_{hor}$ feature vector is 3NK. In this work, $N = 4$ (number of bins), $K = 4$ (number of zones), hence dimension of ZRRL is 48.

7.5.5 Contour hinge (CH) pdf (*f5*)

It is a contour-based feature descriptor first proposed by Bulacu and Schomaker [44]. CH is obtained by computing joint distribution of orientations (ϕ_1 and ϕ_2) of 2 adjacent hinge legs joined at one common point (hinge point). Hinge leg is a short section of a contour of a word consisting of contiguous contour pixels. Orientation (ϕ) of hinge leg is the angle it makes with horizontal measured counter-clockwise as shown in figure 7.4b for hinge legs AB and BC joined at a hinge point B. Both ϕ_1 and ϕ_2 are in the range $0 - 360^o$. $\phi 1$ and $\phi 2$ jointly capture the curvature extent and orientation of ink trace portion (contour hinge ABC in figure 7.4b). The joint instances of ϕ_1 and ϕ_2 are recorded in a orientation histogram with bin size of 15^o. Redundant features are removed for $\phi_1 < \phi_2$ resulting to final feature

dimension of 300.

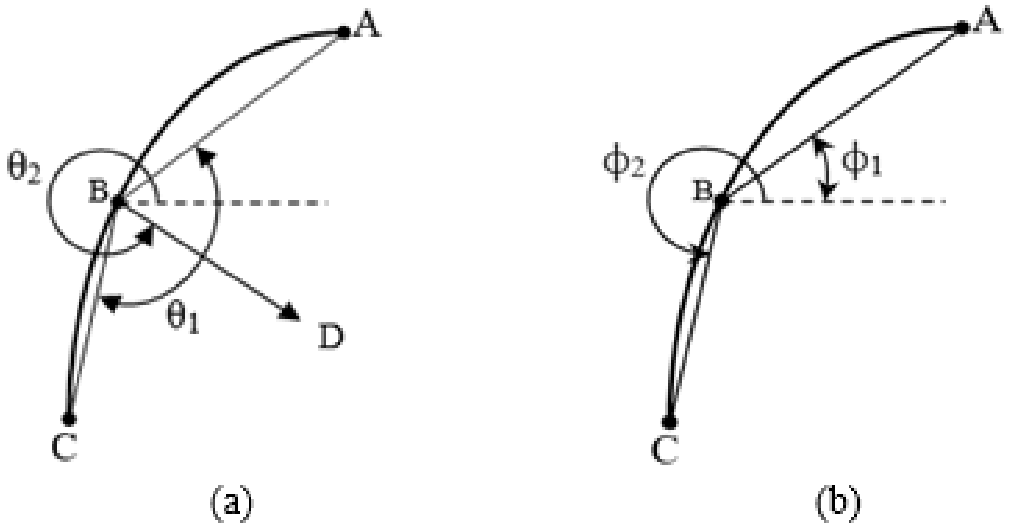


Figure 7.4: Contour hinge formed from 2 fraglets (hinge legs) AB and BC joined at a hinge-point B: (a) illustration of angles θ_1 and θ_2 in modified-CH where BD is bisector of angle ABC on concave side of contour hinge ABC, and (b) illustration of angles ϕ_1 and ϕ_2 in contour hinge pdf [44]

Due to redundancy in CH [44], only features where $\phi_2 > \phi_1$ are considered since they are non-redundant as reported by Bulacu and Schomaker [44]. However, it can be shown that not all the supposedly redundant features (where $\phi_2 < \phi_1$) are actually redundant. The criteria for non-redundancy ($\phi_2 > \phi_1$) depends on designation of ϕ_1 and ϕ_2 which in turn is affected by direction taken when traversing the contour pixels. When traversing through contour pixels, the 1^{st} hinge leg in the traversed direction can be designated as leg 1 and the other as leg 2 or vice versa. Legs 1 and 2 are respectively associated with ϕ_1 and ϕ_2 which are slant angles respective hinge legs make with right-going horizontal line starting at hinge point as shown in figure 7.4b. This can be illustrated with hinges ABC and DEF in a contour shown in figure 7.5. Let's traverse the contour in clockwise direction and designate the AB as leg 1 and BC as leg 2 for hinge ABC (figure 7.5), and DE as leg 1 and EF as leg 2 for hinge DEF (figure 7.5). Contour hinge ABC will be redundant and DEF non-redundant when the non-redundance criteria (where $\phi_2 < \phi_1$) by Bulacu and Schomaker [44], is used. If we reverse the direction of traversing, DEF will be redundant and ABC non-redundant. In

either way one half of contour hinges (ink-traces) will not be considered yet they are crucial in characterising an author's handwriting. The resulting final joint distribution $p(\phi_1, \phi_2)$ in CH [44], is incomplete representation of ink-traces of a given handwriting as shown in figures 7.6a(ii) and 7.6b(ii). The figures 7.6a(ii) and 7.6b(ii) show graphical representations of CH [44], for 2 texts in figures 7.6a(i) and 7.6b(i) respectively written by different writers. They show, entire spectrum of non-redundant (where $\phi_2 > \phi_1$) and redundant (where $\phi_2 < \phi_1$) portions. When only non-redundant region (where $\phi_2 < \phi_1$) is considered, a crucial portion is left out because it is regarded as redundant by Bulacu and Schomaker [44]. Thus, the considered distribution is not a complete reflection of a given handwriting.The consequence is that CH will have a reduced discriminating power compared to when all features are considered in describing ink traces of a handwriting. This is the main weakness of CH [44]. Modified CH (mod-CH) is proposed to address this weakness of CH [44].

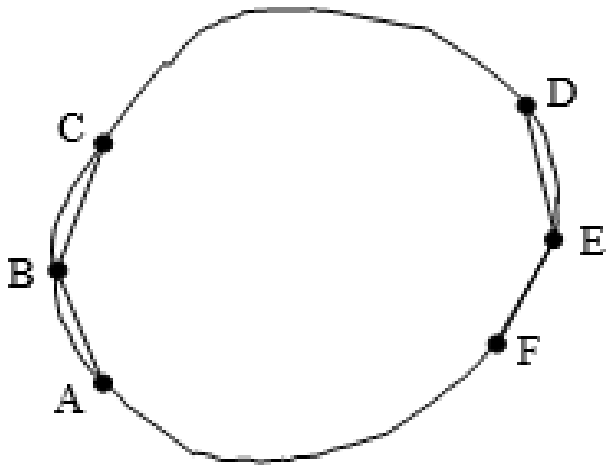


Figure 7.5: Contour of handwritten letter with hinges ABC and DEF

7.5.6 Modified contour hinge (mod-CH) pdf (*f6*)

This is a modified version of contour hinge (CH) feature initially proposed by [44]. It is computed as a joint distribution of hinge's curvature extent (θ_2) and contour hinge orientation (θ_1). Hinge's curvature extent (θ_1) is measured as follows: let AB and BC be 2 adjacent fraglets (hinge legs) making a contour hinge and joined at hinge point B as shown in figure

7.4a. Curvature extent (θ_1) is $\angle ABC$ such that $0 \leq \angle ABC \leq 180^o$ as shown in figure 7.4a. θ_2 is the angle the bisector BD (of θ_1 in figure 7.4a) makes with right going horizontal line starting from B as shown in figure 7.4a. The bisector BD of θ_1 should be on the concave side of contour hinge ABC. Concave side refers to the side of the contour hinge such that $\angle ABC \leq 180^o$. θ_2 is in the range $0 - 360^o$. θ_1 and θ_2 in mod-CH (figure 7.4a) can also be obtained as linear combinations of ϕ_1 and ϕ_2 (figure 7.4b) of CH-pdf [44] using equations 7.4 and 7.5.

$$\theta_1 = \begin{cases} |\phi_1 - \phi_2| & for \ \ |\phi_1 - \phi_2| \leq 180 \\ 360 - |\phi_1 - \phi_2| & Otherwise \end{cases} \tag{7.4}$$

$$\theta_2 = \begin{cases} min\left(\phi_1, \phi_2\right) + |\phi_1 - \phi_2|/2 & for \ \ |\phi_1 - \phi_2| \leq 180 \\ min\left(\phi_1, \phi_2\right) - 180 + |\phi_1 - \phi_2|/2 & for \ \ |\phi_1 - \phi_2| > 180 \ \ \& \ min\left(\phi_1, \phi_2\right) \geq 90 \\ max\left(\phi_1, \phi_2\right) + 180 - |\phi_1 - \phi_2|/2 & for \ \ |\phi_1 - \phi_2| > 180 \ \ \& \ min\left(\phi_1, \phi_2\right) < 90 \end{cases} \tag{7.5}$$

With $0 \leq \theta_1 \leq 180^o$, $0 \leq \theta_2 \leq 360^o$, and θ_1 and θ_2 quantized to bins, the number of combinations computed are $(360/b) * (180/b)$ where b is bin size. Bin size of 15^o is used since it gives sufficient details that describes handwriting styles of authors. Thus, the number of joint bins/combinations is 288. All contour pixels are traversed with each pixel being hinge point. For every hinge point, hinge legs are constituted of 6 contiguous contour pixels on both sides of hinge point. As before, θ_1 and θ_2 are computed for all such contour hinges, and histogram constructed thereof by counting joint instances of θ_1 and θ_2 into their respective joint bins/combinations. Unlike in CH [44], there is no redundance in mod-CH since all joint instances $p\left(\theta_1, \theta_2\right)$ are considered which gives a complete reflection of feature distribution in a given handwriting as shown in figures 7.6a(iii) and 7.6b(iii). This enhances discriminating power of mod-CH as will be shown in section 7.10.4. Mod-CH correlates with contour hinge pdf (CH) feature proposed by Bulacu and Schomaker [44]. θ_1 and θ_2 in mod-CH (figure 7.4a) can be computed from ϕ_1 and ϕ_2 (figure 7.4b) of CH [44] as already mentioned before. Both Mod-CH and CH [44] capture curvature extent and orientation of contour hinges which give more detail about handwriting style of authors. However, they differ in 6 ways: (i) joint

angles capturing curvature extent and orientation of contour hinge in Mod-CH are explicit whereas they are implicit in the case of CH [44], (ii) Mod-CH has no redundance since all its joint instances (288 for bin size of 15^o) are used whereas there is much redundance in the case of CH [44], (iii) Mod-CH considers all ink trace portions whereas CH [44] considers a portion of non-redundant ink traces, (iv) feature space and final feature size of mod-CH is smaller than that of CH [44], (v) feature distribution (i.e., joint distribution of curvature and orientations of contour hinges or ink traces) in CH [44] is affected by direction of traversing through contour pixels whereas direction of traversing does not affect feature distribution in Mod-CH, and (vi) Mod-CH has higher discriminating power than CH [44] as will be shown in section 7.10.4.

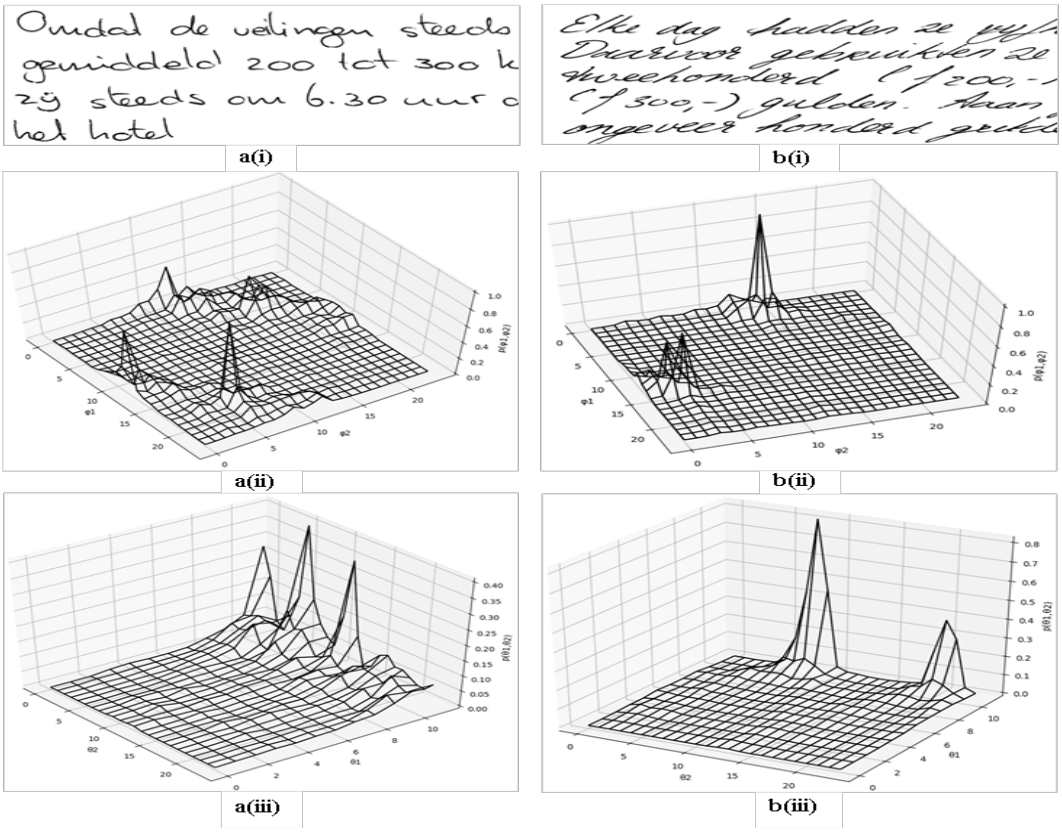


Figure 7.6: Handwritten text of 2 writers in a(i) and b(i) where a(ii) and b(ii) are their respective graphical representations of CH distributions. a(iii) and b(iii) are graphical representations of mod-CH distributions respectively for a(i) and b(i) texts

7.6 Evaluation method for proposed features

The proposed features are evaluated by nearest neighbour matching (NNM) method. NNM is based on distance between words that is computed using features extracted from the words. Words written by same author, under normal conditions and without deliberate faking, have very small distance between them whereas words written by different authors have large distance between them. The distances commonly used are chi-square distance (χ^2) [204], hamming distance, and Euclidean distance. In this work, chi-square distance (χ^2) [204], is used. For 2 words whose feature vectors are represented by x_i and yi, χ^2 distance is computed by equation 7.6.

$$\chi^2 = \sum_{i=1}^{N} \frac{(x_i - y_i)^2}{x_i - y_i} \tag{7.6}$$

Where $i = 1, 2, ..., N$ is index of the dimension respective feature descriptors x and y, and N is the size of feature descriptor.

NNM method for writer identification is suitable where available train data is insufficient for model-based methods, i.e., the number of samples per train class is small. Model-based methods need large train samples per class for efficient trained models to be obtained from training various classifiers like MLP (multi-layer perceptron), SVM (support vector machines), CNN (convolution neural network), and DT (decision trees) among others. Distance based writer identification method is easy to use especially with small data sets. When using NNM, features of a query word (w_q) are matched against features of all words in the database. That is, distances between w_q and all labelled words in database are computed and compared. The author of a word in the database with least distance is deemed to be the author of the query word (w_q). Also, a ranked list of of top k words that are most similar to query word w_q is obtained. In the list, the 1^{st} word is the candidate word with least distance to query word, and k^{th} word is the candidate word with largest distance among the top k similar words.

7.7 Funnelling ensemble method (FEM)

This section discusses the main component of FEM for writer identification that involves computation of meta-features and building meta-classifier. Meta-features for training meta-classifier are generated via k-fold cross validation (KFCV) approach applied to train set. First, train set (Tr) is divided to K disjoint folds $F = \{F_1,\ F_2,\ \ldots\ldots,\ F_K\}$. The splitting is done at class (author) level. That is, words in each train class $Tr_i \in Tr$ (written by single author) are split into K disjoint subsets, i.e., $Tr_i = Tr_{i1}, Tr_{i2},\ \ldots\ldots, Tr_{iK}$ of equal or almost equal sizes. The disjoint subsets are assigned to respective folds. Thus, each train class $Tr_i \in Tr$ is represented in each fold $F_k \in F$ where $k = 1, 2, ..., K$ is fold index.

Thus, train data (Tr) at k^{th} fold (F_k) and y^{th} word (w_y) levels is represented by equations 7.7-7.9.

$$Tr = F = \{F_1,\ F_2,\ \ldots\ldots,\ F_K\} \tag{7.7}$$

$$F_k = \{Tr_{1k},\ Tr_{2k},\ \ldots\ldots, Tr_{Lk}\ \} \tag{7.8}$$

$$Tr_{ik} = \left\{w^i_{1k},\ w^i_{2k},\ \ldots\ldots,\ w^i_{Yk}\right\} \tag{7.9}$$

Where w^i_{yk} represents y^{th} word in k^{th} fold written by i^{th} author, and $k = 1, 2, ..., K$ is fold index, $i = 1, 2, ..., L$ is author index, and $y = 1, 2, ..., Y$ is word index for a subset of words written by a given author in a given fold. Y can vary for different subsets of words in different/same fold/author.

It should be noted that j is word index considering all words written by a given author, and its range is $1 \leq j \leq M$. Therefore, $Y < M$. Therefore, $w^i_{yk} \in Tr_{ik} \in F_k \in F = Tr$. At word level, each fold (F_k) is represented by equation 7.10.

$$F_k = \left\{w^i_{yk}\right\} = \cup^L_{i=1} \cup^Y_{y=1} w^i_{yk} \tag{7.10}$$

Such that $\cup F_k = F = Tr$ and $\cap F_k = \emptyset$ (null set) and i, k, L, y, and Y assume meanings as before.

Base classifier (h_k^1) is trained with words in $\cup_{x\in\{1,2,\ldots K\},x\neq k} F_x$ (that is, $\cup_{x\in\{1,2,\ldots K\},x\neq k}\cup_{i=1}^{L}\cup_{y=1}^{Y} w_{yx}^i$). Base classifier (h_k^1) is used to generate meta-features $(f_k^2(w_u))$ for all $w_u \in F_k$ as shown in equation 7.11. w_u denotes words in fold F_k, which have not been used in training base classifier h_k^1. That is, $w_u = \cup_{x\in\{1,2,\ldots K\},x=k}\cup_{i=1}^{L}\cup_{y=1}^{Y} w_{yx}^i$).

$$f_k^2(w_u) = \left(h_k^1(w_u, a_1),\ h_k^1(w_u, a_2),\ \ldots\ldots\ldots, h_k^1(w_u, a_L)\right) \tag{7.11}$$

Where $h_k^1(w_u, a_i)$ is a numerical classification score that shows the attribution confidence of h_k^1 that word $w_u \in F_k$ is written by i^{th} author (a_i).

Thus, the words used in generating $f_k^2(w_u)$ (equation 7.11) have not been used to train base classifier h_k^1 for k^{th} fold. The same is repeated for all $1 \leq x \leq K$. $f_k^2(w_u)$ from all folds are combined to obtain final output meta feature vector $(f^2(w_u))$ for all words in the train dataset as shown in equation 7.12.

$$f^2(w_u) = \cup_{k=1}^{K} f_k^2(w_u) \qquad \forall\ \ w_u \in F_k \tag{7.12}$$

Meta-classifier (h^2) is trained with meta-feature vectors $f^2(w_u)$ obtained in equation 7.12. The trained meta-classifier (h^2) is used to give final output class (writer) for a test word image (w_t) as explained in next section (section 7.8).

7.8 Getting output class of test word (w_t)

First, base classifiers (h_n^1) used during testing phase are obtained by training feature-specific classifiers with different features of entire train set. That is, for classifier h_n^1, it is trained with feature f_n^1 obtained from entire train set. h_n^1 is different from h_k^1 (section 7.7) in that h_k^1 is obtained with a subset of train set whereas h_n^1 is obtained with entire train set for a given raw feature f_n^1. In obtaining writer of a test word (w_t), a raw feature (f_n^1) is first extracted from it. $f_n^1(w_t)$ should be one of the features used to train base classifiers (h_n^1). The feature

is then fed to respective base classifier (h_n^1) so as to generate L-dimensional vector outputs $f_n^2(w_t)$ as shown in equation 7.13.

$$f_n^2(w_t) = \left(h_n^1(w_t, a_1),\ h_n^1(w_t, a_2),\ \ldots\ldots\ldots, h_n^1(w_t, a_L)\right) \tag{7.13}$$

$f_n^2(w_t)$ is the meta-feature for w_t. $f_n^2(w_t)$ is fed to trained meta-classifier (h^2) to obtain final output class for w_t. The output is L-dimension vector of probabilities showing the confidence the classifier has that input test word w_t belongs to respective classes (representing individual authors) where L is the number of classes (authors). The author of w_t is author corresponding to the class with maximum probability score. The same is repeated for all features used in training base classifiers (h_n^1) and respective classification outputs obtained.

To obtain final classification output (author) due to all features, 2 aggregation schemes are employed: voting and simple averaging schemes. In voting scheme, predicted authors of a test word image (w_t) are obtained for various features *f1-f7*. In case of a tie, the class with highest probability value is the author of w_t. In simple averaging scheme, for every test word image (w_t), vector outputs of probabilities of meta-classifier (h^2) due to various features ($f_n^2(w_t)$) are averaged class-wise as shown in equation 7.14

$$a(w_t) = \cup_{i\in\{0,1,\ldots,L-1\}} \frac{1}{N} \sum_{n=0}^{N-1} h^2\left(f_n^2(w_t), a_i\right) \tag{7.14}$$

where $i = 0, 1, ..., L-1$ is author index where L is number of classes (authors), $n = 0, 1, ..., N-1$ is feature/classifier index where N is number of features/classifiers, and $f_n^2(w_t)$ is vector of meta-features computed by equation 7.11. The author of w_t is one corresponding to class with highest probability in L-dimensional vector $a(w_t)$.

7.9 FEM and Stacking ensemble technique

Stacking ensemble method [205, 206] and FEM are very similar in that both use multiple base (level-1) models whose diverse outputs are combined by high-level meta classifier to get improved overall performance. In both ensemble methods, outputs of base models are

used to develop meta-classifier or meta-model that gives final output. However, there are 3 main distinctions between the two ensemble methods: (i) in stacking ensemble method, same feature is used to build all base models from different classifier types whereas in FEM, base models/classifiers are feature-specific. That is, different features are used to build different base models. (ii) In Stacking, the classifier types used to build base models must be different, whereas in FEM, the classifiers used to build base models need not be different. They may or may not be different since what matters is feature types which must be different for each meta-model. Thus, in FEM, the different feature spaces in level 1 are funnelled/channelled to common feature-space in level 2. (iii) In Stacking, the number of meta-models used is a choice of user but in FEM, the number of meta-models used is same as the number of feature-types used.

7.10 Results and Discussion

This section presents and discusses performances of the designed feature descriptors and the proposed FEM-WI method. First, evaluation data sets and evaluation metrics are presented.

7.10.1 Evaluation datasets

Two datasets of handwritten text documents are used to evaluate the proposed features and FEM-WI method. They are IAM [207] and FireMaker [166] datasets. FireMaker dataset [166]) contains 1004 pages written by 251 students where each student wrote four pages. The 1^{st} page consists of author's normal handwriting in lower caps copied from a standard template, 2^{nd} page is written with author's normal handwriting in upper caps copied from a standard template given. Page 3 consists of forged handwriting of author where each author was asked to deliberately change his/her handwriting style. Page 4 is normal handwriting of authors where they described a given cartoon in their own words. In evaluating FEM-WI method, a set of 1^{st} page of authors is used as train set, and set of 4^{th} page of authors used as test set. IAM dataset [207] consists of 657 writers. It is split into train, test, and valid sets at sentence and word levels. In this work, the set of segmented words is used.

7.10.2 Evaluation metrics

Evaluation metrics used are top-k measures and identification accuracy. For top-k measures, values of k used are 1 and 10. Top-1 and top-10 measures are obtained from top k ranked list of most similar authors obtained with NNM method as explain in section 7.6. The ranked list is ordered from most similar (1^{st}) to least similar (last/k^{th}) words to query word.

(i) *Top-1 measure.* This is the proportion of query words whose correct author appears 1^{st} in the top-k ranked list of similar words together with their authors.

(ii) *Soft Top-10 measure.* This is the proportion of query words whose author appears in the top 10 ranked list of similar words together with their authors.

7.10.3 Evaluation performances of proposed features

IAM [207] and FireMaker [166] data sets have been used to evaluate performance of the designed features. For FireMaker dataset [166], its lower-case sub-sets (pages 1 and 4 sets) are used. The page-wise dataset in FireMaker [166] is segmented to words using method by Omayio et al [201]. The contour hinge PDF feature is included since it is the best performing feature by Bulacu and Schomaker [44]. Chi-square distance [204] is used in comparing query word image with candidate word images in respective data sets. The features are evaluated individually and as combinations amongst themselves. Combinations are done to enhance performance.

7.10.4 Performance of individual proposed features

Tables 7.2 shows performances of the individual features with the 2 mentioned evaluation datasets using top-1 and top-10 metrics. The measures are computed using NNM as explained in sections 7.6 and 7.10.2. From the table, it can generally be seen that performance of the features is consistent for the 2 datasets used. The best performing feature is mod-CH (*f6*) followed by FHOD (*f2*). mod-CH (*f6*) has the best performance because it captures more details resulting from its joint distribution of hinge concave angle (θ_2) and orientation (θ_1)

of concave side of the hinge (section 7.5.6). FHOD (*f2*) also has good performance because it captures local curvature information of fraglets and oriented displacement of fraglet (pixel) points with respect to center of bounding box of the fraglet. This characterises well the ink traces for a given handwriting. CH (*f5*) [44] though with similar information as mod-CH (*f8*), its performance is less than that of mod-CH (*f6*). This is because in a bid to avoid redundance, some essential features are left out which reduces it discriminating power as explained before (in section 7.5.5). ZRRL-vert (*f4v*) has good performance because it contains information of run length with regards to adjacent run lengths and position of white runs which uniquely characterises author's writing style. The least performing feature is RRL_{hor} (*f3h*) due to less information in contains relative to other features. It can also be noted that mod-CH outperforms CH [44] for both IAM [207] and FireMaker [166] data sets. This is because mod-CH has no redundance and considers all contour hinges during extraction (section 7.5.6) in describing a handwriting whereas CH (*f5*) [44] does not since redundant features are left out that eventually reduces its discriminating power.

Table 7.2: Performance of proposed features with IAM[207] and FireMaker[166] datasets with NNM

	Feature	Dimension	IAM[207]		Firemaker[166]	
			Top-1	Top-10	Top-1	Top-10
f1	HOO	24	88	93	86	90
f2	FHOD	24	93	95	89	94
f3h	RRL-hor	12	82	86	83	86
f3v	RRL-vert	12	85	89	84	89
f4h	ZRRL-hor	48	83	89	83	87
f4v	ZRRL-vert	48	89	94	87	91
f5	CH [44]	300	81	92	81	92
f6	Mod-CH	288	94	97	90	96

7.10.5 Performance of combined proposed features

The combination scheme used is by concatenating individual features for each word image to form one aggregate feature. Performance of the aggregate feature is then evaluated using NNM (section 7.6) with top-1 and top-10 measures (section 7.10.2). Tables 7.3 shows performances of various features combinations for IAM [207] and FireMaker [166] data sets.

From the table, it can be seen that for all feature combinations, improved performance is obtained compared to individual features. Also, there is consistent performance of all feature combinations across the 2 datasets used for evaluation. The best performing combination is $f1+f2+f4h+f4v+f6$ followed by $f2+f6$. The least performing combination is that of $f3h+f3v$. Different features capture different informations from handwritten word images, and have different strengths and weaknesses. As a result, more and variety of information is captured, and complementariness achieved gives improved performance. In spite of the improved performance of feature combinations, they have increased feature dimension, hence high computational cost.

Table 7.3: Performance of combined features with IAM[207] and FireMaker[166] datasets

Feature combination	Dimension	IAM [207]		Firemaker [166]	
		Top1	Top10	Top1	Top10
f3h+f3v	24	88	94	86	93
f4h+f4v	96	94	98	91	96
f2+f5	324	91	97	92	96
f4h+f4v+f6	384	92	97	90	94
f1+f6	312	94	97.5	91	96.5
f2+f6	312	96	98	93	97
f1+f4h+f4v+f6	408	94	97.5	90	97
f1+f2+f4h+f4v+f6	432	96	98.5	97	98.6
f4h+f4v+f5	396	94	97	91	96

7.11 Performance of FEM-WI technique

In evaluating the proposed FEM-WI method, a set of 5 features were used (2 individual features and 3 sets of combined features). Individual features used are *f2* and *f6*, and feature combinations used are *f1+f6*, *f2+f6*, and $f1+f2+f4h+f4v+f6$. The features are used due to their good performances and relatively low feature dimension. They are used to train feature-specific base classifiers (sections 7.4.2). The base classifiers are in turn used to compute meta-features (sections 7.4.3 and 7.7). The meta-features are then used to train meta-classifier that outputs final class (author) for a given input test word image (section 7.8). In this work, multi-layer perceptron (MLP) classifier was used in both base-classifiers

and meta-classifiers. For base classifiers, 3-layered MLP was used where 1^{st} and 2^{nd} layers consist of 2048 and 1024 units respectively. Drop-out rate of 0.4 was used to minimize overfitting. ReLU (Rectified linear unit) activation function was used in layers 1 and 2. In the 3^{rd} (last) layer, activation used is softmax. Adaptive moment (Adam) optimizer [123] was used during MLP trainings process.

Table 7.4 shows performance of FEM-WI method with various individual features and feature combinations. The features are input to the respective base classifiers (h_n^1). The meta-features obtained are then fed to meta-classifier (h^2) to obtain final output vector of probabilities. From table 7.4, performance improvement can be seen in all features when FEM-WI method is used as compared to nearest neighbour method (section 7.6). For instance, feature *f2* attains top-1 and top-10 writer identification rates of 98.5% and 99% respectively for IAM dataset [207]. The same feature (*f2*) attains top-1 and top-10 writer identification rates of 93% and 95% respectively for IAM dataset [207] when nearest neighbour method (NNM) is used for evaluation (see table 7.2). A similar performance improvement is also attained for feature *f2* when evaluated with FireMaker [166] data set FEM-WI method. Similar cases of performance improvement are obtained for other features as shown in table 7.4. This shows that FEM-WI method is very efficient for writer identification, resulting from efficient combination scheme of base classifier outputs for final classification, realized by meta-classifier (h^2). With FEM-WI method, the more the number of feature-specific base classifiers used, the more the number of meta-features available for training meta-classifier, and hence a more efficient meta-classifier obtained as compared to individual base classifiers. This results to improved overall performance.

The final classification outputs of meta-classifier for individual features are aggregated by voting and averaging schemes as explained in section 7.8. The performance of FEM-WI method with aggregated outputs is compared with state-of-the-art writer identification methods as shown in table 7.5. In aggregating final outputs for various features, it can be seen that aggregation by averaging scheme performs better than by voting scheme. For IAM dataset [207] for instance, top-1 writer identification rates for voting and averaging schemes are 98.4% and 99% respectively. From table 7.5, it can also be seen that for IAM dataset [207],

the proposed method (with averaging scheme) has best writer identification rates of 99.0% and 99.6% respectively for top-1 and top-10 metrics, followed by that of Wu et al [208]. For FireMaker dataset [166], the proposed method has best top-1 writer identification rate of 95% followed closely by that of Wu et al [208]. For top-10 metric, the proposed method is best with 99.1%. This shows that FEM-WI with averaging scheme is efficient in writer identification. This is because during class-wise averaging for various features (section 7.8), features complement one another in that strength of one feature makes for the weakness of another. This results to improved and higher performance.

Table 7.4: Performance of the proposed FEM-WI method with various features for IAM[207] and FireMaker[166] datasets

Feature	Dimension	IAM [207]		Firemaker[166]	
		Top-1	Top-10	Top-1	Top-10
f2	24	98.5	99	93	97.5
f6	288	97	98	94	96
f1+f6	312	98.8	99.5	94	97
f2+f6	312	96	98	92	97
f1+f2+f4h+f4v+f6	396	99.0	99.6	95	99.1

7.12 Conclusion

In this chapter, an efficient funnelling ensemble method (FEM) for independent handwritten writer identification has been presented. It is an ensemble method consisting of 2 main levels of classifiers: base and meta-classifiers. Base classifiers are feature-specific, and are built by training MLP with different feature-types. The base classifiers are used to generate meta-features which in turn are used to train meta-classifier. Meta-classifier gives final output of author-class for a given test word image. Also, 5 new feature descriptors have been proposed that capture characteristic handwriting styles of different authors. The features are, Histogram of orientation (HOO), Fraglet histogram of oriented displacement (FHOD), Relative run length (RRL), Zoned Relative run length (ZRRL), and modified contour hinge (mod-CH). The features have high discriminating powers, and are size and scale invariant making them efficient for writer identification tasks. The proposed features and FEM-WI method

Table 7.5: Performance comparison of FEM-WI method for aggregated meta-classifier outputs with state-of-the-art writer identification methods for IAM [207] and FireMaker [166] datasets

Writer identification method	Feature	IAM[207]		Firemaker[166]	
		Top1	Top10	Top1	Top10
Bulacu and Schomaker [44]	Contour hinge	81.0**	92.0**	81.0**	92.0**
He and Schomaker [52]	Fraglets	72.2	88.0*	75.9	94.7*
Brink et al [45]	Quill–Hinge	97.0	98.0	86.0	97.0
Brink et al [45]	Quill	95	97	71	89
Lai and Jin [46]	Local path signature	94.24	97.77	91.0	97.0
Wu et al [208]	SIFT features	98.5	99.5	92.4	98.8
Ghiasi and Safabakhsh [209]	OHS	93.7	97.7	91.8	98.6
He and Schomaker [193]	Delta-n Hinge	93.2	97.2	90.4	98.2
Khalifa et al [22]	GCF	92	-	-	-
Siddiqi and Vincent [211]	USC	84	96	-	-
Tang et al [212]	SFH	96.7	97.7	88.2	96.2
Tang et al [212]	LCPH	94.0	96.1	78.8	92.2
Proposed (voting scheme)	*f1+f2+f4h+f4v+f6*	98.4	99.1	91.5	96.5
Proposed (averaging scheme)	*f1+f2+f4h+f4v+f6*	99.0	99.6	95.0	99.1
* Top-5 measure; ** Lower case set was used OHS - occurrence histogram of the shapes; GCF - grapheme codebook features USC-Universal shape codebook SIFT - scale invariant feature transform; SFH-Stroke Fragment Histogram LCPH - Local Contour Pattern Histogram					

have been evaluated with IAM [207] and FireMaker [166] datasets of handwritten text documents. The top 2 features are modified contour hinge (mod-CH) and fraglet histogram of oriented displacement (FHOD). When evaluated with IAM and Firemaker datasets, mod-CH attained top-1 measures of 94% and 90% respectively. FHOD attained top-1 measures of 93% and 89% respectively for IAM [207] and FireMaker [166] datasets. The least performing feature descriptor is RRLhor with top-1 measures of 82% and 83% respectively for IAM [207] and FireMaker [166] datasets. The proposed FEM-WI posted top-1 identification rate of 99.0% and 95.0% respectively for IAM [207] and FireMaker [166] with simple averaging scheme.

Chapter 8

CONCLUSION AND RECOMMENDATIONS

In this thesis, various techniques have been formulated for various historical manuscript processing and management tasks. Binarization and word segmentation techniques have been used to prepare historical manuscript images (HMI) for other manuscript processing tasks like writer identification, word spotting, and language identification among others. Other techniques developed are for handwritten word spotting (HWS), language identification (LID), writer identification, and era/creation time prediction. These methods have been employed for historical manuscript classification. All these methods have been objectively/quantitatively evaluated and also compared with respective state-of-the-art methods handling same tasks. The proposed techniques have desirable output performances.

8.1 Summary of work done in this thesis

Due to degradations of various forms found in historical manuscripts, multilayer perceptron (MLP)-based binarization technique was developed to binarize HMIs. This technique models HMI pixels to foreground and background by training MLP classifier using 5 features: per-region fuzzy membership grades, local contrast, local standard deviation, entropy feature, and normalized gray levels. The method is able to obtain good quality binary images

especially from HMI with degradations like ink bleeds, see throughs, fading, and uneven illumination.

In addressing the challenges of segmentation of crossing and overlapping words, component tracing and association (CTA) technique was developed to handle the same. By tracing strokes of a word, full word segmentation is made possible even for strokes crossing with those from adjacent words. Crossing strokes are separated by using junction branch association (JBA) algorithm that uses multi-dimensional dynamic time warping with dependence ($MD - DTW_D$) method. High performance was achieved with this method especially in segmentation of crossing and overlapping words.

Integral histogram of oriented displacement (IHOD) descriptor was designed for use in developing handwritten word spotting (HWS) model by training MLP classifier. IHOD descriptor consists of global and local information of segmented handwritten word image, thus having high discriminating power. The HWS model was used for indexing HMI.

Fragmented LSTM for language identification (FragLSTM-LID) framework was developed for LID of textual content of HMI. This is made possible by learning and extracting global and local information from text words using 3 LSTM networks. This LID approach was used for classification of HMI. High classification accuracy scores were achieved.

In predicting era or production time of HMI, bi-directional fragmented network (BiD-FragNet) technique was developed. This method uses 2 CNN channels to learn and extract local and global features from patches of HMIs. These features are in turn used with classification layer to estimate era/creation time of HMIs. Good results were obtained.

Finally, funelling ensemble method for writer identification (FEM-WI) was developed. In developing the method, 5 new feature descriptors were proposed: Histogram of orientation (HOO), Fraglet histogram of oriented displacement (FHOD), Relative run length (RRL), Zoned Relative run length (ZRRL), and modified contour hinge (mod-CH). FEM-WI consists of 2-level system of classifier ensembles. The features are used to develop level 1 base classifiers. The features are funnelled to a common feature space using level 1 classifiers. Meta features are obtained as outputs of base classifiers which in turn are used to train level 2 meta classifier which gives final output classification. In this way, when one feature is used

to predict writer of a given manuscript document, it benefits from all other features use to train meta classifier. FEM-WI gives very good classification results.

8.2 Contributions

The computer-based methods developed in this thesis help in various historical manuscript management tasks. The contributions made in this thesis are as follows:

(i) Design of MLP-based binarization of HMI. The proposed technique binarized well degraded HMI.

(ii) Design of IHOD descriptor for handwritten key-word spotting.

(iii) Development of an efficient word segmentation technique that handles well overlapping and crossings in handwritten words.

(iv) Developed LSTM-based language identification technique for historical manuscript texts.

(v) Designing of deep learning-based method for era or production time prediction for HMIs.

(vi) Designed funelling ensemble method for manuscript writer identification.

8.3 Future scope

(i) Create a large database of under-resourced languages to help develop efficient language identification systems.

(ii) Develop a method for handling torn and incomplete HMIs.

(iii) Explore the use of more than 2 channels in BiD-FragNet system for era prediction of HMIs.

PUBLICATIONS

(a) **Publications in Journals**

(i) **Enock Osoro Omayio, Indu Sreedevi, Jeebananda Panda (2022). Word Segmentation by Component Tracing and Association (CTA) Technique. Journal of Engineering Research. Doi: https://doi.org/10.36909/jer.15207**

(ii) **Enock Osoro Omayio, Indu Sreedevi, Jeebananda Panda (2022). Historical manuscript dating: traditional and current trends. Multimedia Tools and Applications. doi: https://doi.org/10.1007/s11042-022-12927-8**

(b) **Publications in International conference**

(i) **Enock Osoro Omayio, Indu Sreedevi, Jeebananda Panda (2021). Word spotting of handwritten Hindi scripts by circular histogram of oriented displacement (CHOD) features. 4th Biennial International Conference on Nascent Technologies in Engineering, 15-16 Jan, 2021. Fr. C. Rodrigues Institute of Technology, Vashi, Navi Mumbai, India. Doi: 10.1109/ICNTE51185.2021.9487701**

(ii) **Enock Osoro Omayio, Indu Sreedevi, Jeebananda Panda (2022). Multilayer perceptron model for document image binarization. 38th World Conference on Applied Science, Engineering & Technology (WCASET), 27-28th October, 2021, Manila, Philippines.**

(c) **Book Chapters published**

(i) **Enock Osoro Omayio, Indu Sreedevi, Jeebananda Panda (2021). Introduction to Heritages and Heritage Management: A Preview. Digital Techniques for Heritage Presentation and Preservation. Springer. ISBN 978-3-030-57906-7. ISBN 978-3-030-57907-4 (eBook). Doi: https://doi.org/10.1007/978-3-030-57907-4.**

(ii) **Enock Osoro Omayio, Indu Sreedevi, Jeebananda Panda (2021). Language-Based Text Categorization: A Survey. Digital Techniques for Heritage Presentation and Preservation. Springer. ISBN 978-3-030-57906-7. ISBN 978-3-030-57907-4 (eBook). Doi: https://doi.org/10.1007/978-3-030-57907-4**

(d) **Papers communicated to International Journals**

(i) **Enock Osoro Omayio, Indu Sreedevi, Jeebananda Panda (2021). Word spotting and character recognition of handwritten Hindi scripts by Integral Histogram of Oriented Displacement (IHOD) descriptor. Multimedia Tools and Applications. (Under review)**

(ii) **Enock Osoro Omayio, Indu Sreedevi, Jeebananda Panda (2022). BiD-FragNet: Manuscript dating using Bi-directional fragment networks. Journal of Computer Science and Technology. (Under review).**

REFERENCES

[1] Adam, K., Al-Maadeed, S., and Bouridane, A. (2017). Letter-based classification of Arabic scripts style in ancient Arabic manuscripts. IEEE International Workshop on Arabic Script Analysis and Recognition (ASAR), pp. 95-98.

[2] Hamid, A., Bibi, M., Moetesum, M., and Siddiqi, I. (2019). Deep learning based approach for historical manuscript dating. Proceedings of the International Conference on Document Analysis and Recognition (ICDAR), pp. 967-972. https://doi.org/10.1109/ICDAR.2019.00159

[3] Nesměrák, K., and Němcová, I. (2012). Dating of Historical Manuscripts Using Spectrometric Methods: A Mini-Review, Analytical Letters, 45(4):330- 344. doi: 10.1080/00032719.2011.644741

[4] Metzger, B.M. (1981). Manuscripts of the greek bible: an introduction to greek paleography. Oxford 1122 University Press, Inc., pp. 14.

[5] Omayio, E.O., Indu, S., and Panda, J. (2022). Historical manuscript dating: traditional and current trends. Multimedia Tools and Applications. doi: https://doi.org/10.1007/s11042-022-12927-8.

[6] Otsu, N. (1979). A threshold selection method from grey level histogram. IEEE Trans. Syst. Man Cybern. 9:62-66.

[7] Xueting, H., Shiqian, W., and Wangming, X. (2019). Adaptive binarization for degraded document image via contrast enhancement. Proc. 14th IEEE Conf. Industrial Electronics and Applications, pp. 2362-2367.

[8] Niblack, W. (1986). An Introduction to Digital Image Processing. Englewood Cliffs NJ, Prentice Hall, pp. 115-116.

[9] Jayanthi, N. and Indu, S. (2017). Enhancement of ancient manuscript images by log based binarization technique. Int. Journal of Electronics and Communication, 75:15-22.

[10] Khurshid, K., Siddiqi, I., Faure, C., and Vincent, N. (2009). Comparison of Niblack inspired Binarization methods for ancient documents. Proc. 16th Int. conf. Document Recognition and Retrieval, USA, 2009.

[11] Sauvola, J. and Pietikainen, M. (2000). Adaptive document image binarization. Pattern Recognition, 33:225-236.

[12] Lu, D., Huang, X., and Sui, L. (2018). Binarization of degraded document images based on contrast enhancement. Int. Journal on Document Analysis and Recognition, 21:123-135.

[13] Singh, B.M. Sharma, R., Ghosh, D., and Mittal, A. (2014). Adaptive binarization of severely degraded and non-uniformly illuminated documents. Int. Journal on Document Analysis and Recognition, 17:393-412.

[14] Bernsen, J. (1986). Dynamic thresholding of gray-level images. Proc. Int. Conf. Pattern Recognition, pp. 1251-1255.

[15] Gatos, B., Pratikakis, I., and Perantonis, S.J. (2006). Adaptive degraded document image binarization. Pattern Recognition, 39:317-327.

[16] Vinod, H.C. and Niranjan, S.K. (2018). Binarization and Segmentation of Kannada Handwritten Document Images. Proc. 2nd Int. conf. Green Comput. Intern. Things, IEEE, pp. 488-493.

[17] Papamarkos, N. (2003). A neuro-fuzzy technique for document binarization. Neural Comput. Appl., 12(3-4):190-199.

[18] Mitianoudis, N. and Papamarkos, N. (2015). Document image binarization using local features and Gaussian mixture modelling. Image and Visual Computing, Elsevier, 38:33-51.

[19] Howe, N.R. (2011). A Laplacian energy for document binarization. Proc. Int. Conf. Document Analysis and Recognition, IEEE, pp. 6-10.

[20] Wu, S. and Amin, A. (2003). Automatic thresholding of gray-level using multistage approach. Proc. 7th Int. Conf. Document Analysis and Recognition, 2003.

[21] Tanaka, H. (2009). Threshold Correction of Document Image Binarization for Ruled-line Extraction. Proc. 10th Int. Conf. on Document Analysis and Recognition. pp. 545.

[22] Kefali, A., Sari, T., and Bahi, H. (2014). Foreground-Background Separation by Feed-forward Neural Networks in Old Manuscripts. Informatica, 38:329-338.

[23] Wu, Y., Natarajan, P., Rawls, S., and AbdAlmageed, W. (2016). Learning document image binarization from data. IEEE Int. Conf. Image Processing, pp. 3763-3767.

[24] He, S. and Schomaker, L. (2019). DeepOtsu: Document Enhancement and Binarization using Iterative Deep Learning. Journal of Pattern Recognition, 91:379-390.

[25] Afzal, M.Z. et al. (2015). Document image binarization using lstm: a sequence learning approach. Proc. 3rd Int. Workshop on Historical Document Imaging and Processing. ACM, pp. 79-84.

[26] Tensmeyer, C. and Martinez, T. (2017). Document Image Binarization with Fully Convolutional Neural Networks. 14th IAPR Int. Conf. Document Analysis and Recognition, pp 99-104.

[27] Peng, X., Cao, H., and Natarajan, P. (2017). Using Convolutional Encoder-Decoder for Document Image Binarization. 14th IAPR Int. Conf. Document Analysis and Recognition, pp. 708-713.

[28] Akbari, Y., Britto, A.S., Al-maadeed, S., and Oliveira, L.S. (2019). Binarization of Degraded Document Images using Convolutional Neural Networks based on predicted Two-Channel Images. Int. Conf. Document Analysis and Recognition, pp. 973-978.

[29] Fischer, A., Keller, A., Frinken, V., and Bunke, H. (2012). Lexicon-free handwritten word spotting using character HMMs. Pattern Recognition Letters, 33(7):934–942.

[30] Sudholt, S. and Fink, G.A. (2016). PHOCNet: A deep convolutional neural network for word spotting in handwritten documents. Proceedings of the 15th Int. Conf. Frontiers in Handwriting Recognition (ICFHR), Shenzhen, China, pp. 277-282.

[31] Arora, S., Bhattacharjee, D., Nasipuri, M., Malik, L., Kundu, M., and Basu, D.K. (2010). Performance comparison of SVM and ANN for handwritten Devnagari character recognition. International Journal of Computer Science Issues, 7(3):18-26.

[32] Retsinas, G., Louloudis, G., Stamatopoulos, N., and Gatos, B. (2019). Efficient Learning-Free Keyword Spotting. IEEE transactions on pattern analysis and machine intelligence, 41(7):1587-1600.

[33] Bhardwaj, A., Jose, D., and Govindaraju, V. (2008). Script Independent Word Spotting in Multilingual Documents. Proceedings of the 2nd workshop on Cross Lingual Information Access (CLIA) Addressing the Information Need of Multilingual Societies, pp 48-54.

[34] Srihari, S.N., Srinivasan, H., Huang, C., and Shetty, S. (2006). Spotting words in Latin, Devanagari and Arabic scripts. Vivek: Indian Journal of Artificial Intelligence, 6(3):2-9.

[35] Shekhar, R. and Jawahar, C.V. (2012). Word Image Retrieval using Bag of Visual Words. 10th IAPR International Workshop on Document Analysis Systems, IEEE pp. 297-301.

[36] Hassan, E., Chaudhury, S., and Gopal, M. (2013). Word shape descriptor-based document image indexing: A new DBH-based approach. International Journal on Document Analysis and Recognition, 16:227-246.

[37] Zhang, X., Pal, U., and Tan, C.L. (2014). Segmentation-free Keyword Spotting for Bangla Handwritten Documents. In Proc. International Conference on Frontiers in Handwriting Recognition, pp. 381-386.

[38] Zagoris, K., Pratikakis, I., and Gatos, B. (2014). Segmentation-based Historical Handwritten Word Spotting using Document-Specific Local Features. 14th International Conference on Frontiers in Handwriting Recognition, IEEE, pp. 9-14. doi: 10.1109/ICFHR.2014.10.

[39] Roy, P.P., Bhunia, A.K., Das, A., Dey, P., Pal, U. (2016). HMM-based Indic Handwritten Word Recognition using Zone Segmentation. Pattern Recognition, 60:1057-1075,

[40] Bhunia, A. K., Roy, P.P., Sain, A., and Pal, U. (2020). Zone-based keyword spotting in Bangla and Devanagari documents. Multimedia Tools and Applications, 79:27365–27389, doi: https://doi.org/10.1007/s11042-019-08442-y.

[41] Bensefia, A. and Djeddi, C. (2020). Relevance of grapheme's shape complexity in writer verification task. IEEE 21st International Conference on Information Reuse and Integration for Data Science (IRI), pp. 53-58.

[42] Newell, A.J. and Griffin, L.D. (2014). Writer identification using oriented basic image features and the delta encoding. Pattern Recognit., 47(6):2255-2265.

[43] Kuhl, F. P. and Giardina, C.R. (1982). Elliptic Fourier features of a closed contour. Comput. Graph. Image Process., 18(3):236-258, doi:https://doi.org/10.1016/0146-664X(82)90034-X.

[44] Bulacu, M. and Schomaker, L. (2007). Text-independent writer identification and verification using textural and allographic features. IEEE Trans. Pattern Anal. Mach. Intell., 29(4):701-717.

[45] Brink, A., Smit, J., Bulacu, M., and Schomaker, L. 2012. Writer identification using directional ink-trace width measurements, Pattern Recognit., 45 (1):162–171.

[46] Lai, S. and Jin, L. (2019). Offline Writer Identification based on the Path Signature Feature, arXiv:1905.01207v1 [cs.CV].

[47] Chahi, A., El Merabet, Y., Ruichek, Y., and Touahni, R. (2019). Off-line Text independent Writer Identification Using Local Convex Micro-Structure Patterns. In New Challenges in Data Sciences: Acts of the Second Conference of the Moroccan Classification Society (SMC '19), KENITRA, Morocco, https://doi.org/10.1145/3314074.3314080

[48] He, S. and Schomaker, L. (2017). Writer identification using curvature-free features. Pattern Recognition, 63:451-464.

[49] Nicolaou, A., Bagdanov, A.D., Liwicki, M., and Karatzas, D. (2015). Sparse radial sampling lbp for writer identification. International Conference on Document Analysis and Recognition (ICDAR), pp. 716-720.

[50] Bertolini, D., Oliveira, L. S., Justino, E., and Sabourin, R. (2013). Texture-based descriptors for writer identification and verification. Expert Systems with Applications, 40 (6):2069-2080.

[51] Krizhevsky, A., Sutskever, I., and Hinton, G.E. (2012). ImageNet classification with deep convolutional neural networks. Proceedings of the Adv. Neural Inf. Process. Syst., pp. 1097-1105.

[52] He, S. and Schomaker, L. (2020). FragNet: Writer Identification Using Deep Fragment Networks. IEEE Transactions on Information Forensics and Security, 15:3013-3022.

[53] Xing, L. and Qiao, Y. (2016). DeepWriter: A Multi-Stream Deep CNN for Text-independent Writer Identification. 15th International Conference on Frontiers in Handwriting Recognition, pp. 584-589.

[54] Cilia, N.D., De Stefano, C., Fontanella, F., Marrocco, C., Molinara, M. and Di Freca, A.S. (2020). An end-to-end deep learning system for medieval writer identification. Pattern Recognition Letters, 129:137–143.

[55] Nguyen, H.T., Nguyen, C.T., Ino, T., Indurkhya, B. and Nakagawa, M. (2019). Text-independent writer identification using convolutional neural network. Pattern Recognition Letters, 121:104-112.

[56] Oda, H., Ikeda, K. (2010). Radiocarbon dating of kohitsugire calligraphies attributed to Fujiwara Shunzei: Akihiro-gire, Oie-gire, and Ryosa-gire. Nucl. Instr. and Meth. in Phys. Res. B, 268:1041-1044.

[57] Jull, A. J. T., Donahue, D.J., Broshi, M., and Tov, E. (1995). Radiocarbon dating of scrolls and linen fragments from the Judean desert. Radiocarbon, 37: 11–19.

[58] Larkin, P. (2011). Infrared and Raman Spectroscopy: Principles and Spectral Interpretation, Elsevier Inc., 225 Wyman Street, USA.

[59] Gauglitz, G., and Vo-Dinh, T. (2003). Handbook of Spectroscopy. WILEY-VCH Verlag GmbH & Co. KGaA, Weinheim.

[60] Jull, A.J.T. and Burr, G. S. (2014). Some interesting applications of radiocarbon dating to art and archaeology. Archeometriai Muhely, 11(3):139-148,

[61] Jull, A.J.T. (2013). Some interesting and exotic applications of carbon-14 dating by accelerator mass spectrometry. J. Phys.: Conf. Ser. 436:012083. doi:10.1088/1742-6596/436/1/012083.

[62] He, S., Samara, P., Burgers, J., and Schomaker, L. (2016a). Image-based historical manuscript dating using contour and stroke fragments. Pattern Recognition, 58:159-171.

[63] Garain, U., Parui, S.K., Paquet, T., and Heutte, L. (2007). Machine Dating of Handwritten Manuscripts. 9th International Conference on Document Analysis and Recognition (ICDAR). doi: 10.1109/ICDAR.2007.4377017.

[64] National Geographic https://www.nationalgeographic.com/news/2013/5/130530-worlds-oldest-torah-scroll-bible-bologna-carbon-dating/. Accessed 20 December 2022

[65] Mani, I. and Wilson, G. (2000). Robust temporal processing of news. In: ACL 2000: Proceedings of the 38th Annual Meeting on Association for Computational Linguistics.

[66] Kim, Y.K. (1988). Palaeographical Dating of P46 to the Later First Century, Biblica, 69(2):248-257.

[67] Llidó, D.M., Llavori, R.B., and Cabo, M.J.A. (2001). Extracting temporal references to assign document event-time periods. In: Mayr HC, Lazansk´y J, Quirchmayr G, Vogel P (eds.) DEXA 2001. LNCS, vol. 2113, Springer, Heidelberg.

[68] Sanders, H.A. (1935). A Third Century Papyrus Codex of the Epistles of Paul. Ann Arbor, Mich., University of Michigan Press.

[69] Wang, X., Feng, B., Bai, X., Liu, W., and Latecki, L.J. (2014). Bag of contour fragments for robust shape classification. Pattern Recognition, 47(6):2116-2125.

[70] Latecki, L. J. and Lakamper, R. (1999) Convexity rule for shape decomposition based on discrete contour evolution. Comput. Vis. Image Underst., 73(3):441-454.

[71] He, S., Wiering, M., and Schomaker, L. (2015). Junction detection in handwritten documents and its application to writer identification. Pattern Recognit., 48(12):4036-4048.

[72] Wahlberg, F., Martensson, L., and Brun, A. (2015). Large scale style-based dating of medieval manuscripts. In: Workshop on Historical Document Image and Processing (HIP), pp. 107-114, doi: http://dx.doi.org/10.1145/2809544.2809560.

[73] Dhali, M. A., Jansen, C. N., Wit, J. M., and Schomaker, L. (2020). Feature-extraction methods for historical manuscripts dating based on writing style development. Pattern Recognition Letters, 131:413-420. doi: 10.1016/j.patrec.2020.01.027.

[74] Al-Aziz, A.M.A., Gheith, M., and Sayed, A.F. (2011). Recognition for old Arabic manuscripts using spatial gray level dependence (SGLD). Egyptian Informatics Journal, 12(1):37-43. Doi: https://doi.org/10.1016/j.eij.2011.02.001.

[75] He, S., Samara, P., Burgers, J., and Schomaker, L. (2016b). Historical manuscript dating based on temporal pattern codebook. Computer Vision and Image Understanding, 152:167-175, doi: http://dx.doi.org/10.1016/j.cviu.2016.08.008.

[76] He, S. and Schomaker, L. (2015). A polar stroke descriptor for classification of historical documents. In International Conference on Document Analysis and Recognition (ICDAR), pp. 6-10.

[77] He, S., Samara, P., Burgers, J., and Schomaker, L. (2014). Towards style-based dating of historical documents. In: International Conference of Frontiers in Handwriting Recognition (ICFHR), pp. 265-270.

[78] Hamid, A., Bibi, M., Siddiqi, I., and Moetesum, M. (2018). Historical Manuscript Dating using Textural Measures. IEEE International Conference on Frontiers of Information Technology (FIT), pp. 235-240. Doi:10.1109/FIT.2018.00048.

[79] Wahlberg, F., Wilkinson, T., Brun, A. (2016). Historical Manuscript Production Date Estimation using Deep Convolutional Neural Networks. 15th International Conference on Frontiers in Handwriting Recognition, pp. 205-210. doi:10.1109/ICFHR.2016.0048.

[80] Svenskt Diplomatariums huvudkartotek (SDHK), https://sok.riksarkivet.se/SDHK, last accessed 10 May, 2022.

[81] Souter, C., Churcher, G., Hayes, J., Hughes, J., Johnson, S. (1994). Natural language identification using corpus-based models. Hermes Journal of Linguistics, 13:183-203.

[82] Dongen, N. (2017). Analysis and Prediction of Dutch-English Code-switching in Dutch Social Media Messages. Master's thesis, Universiteit van Amsterdam, Amsterdam, Netherlands.

[83] Kumar, R.V.R.M., Kumar, A.M., and Soman, K.P. (2015). AmritaCEN NLP @ FIRE 2015 Language Identification for Indian Languages in Social Media Text. In: Working Notes of the Forum for Information Retrieval Evaluation (FIRE 2015), pp. 28–30, Gandhinagar, India.

[84] Clematide, S. and Makarov, P. (2017). CLUZH at VarDial GDI 2017: Testing a Variety of Machine Learning Tools for the Classification of Swiss German Dialects. In: Proceedings of the Fourth Workshop on NLP for Similar Languages, Varieties and Dialects, Valencia, Spain, pp. 170-177.

[85] Saharia, N. (2017). Phone-based Identification of Language in Code-mixed Social Network Data. Journal of Statistics and Management Systems, 20(4):565-574.

[86] Cagigós, S.P. (2017). Catdetect, a framework for detecting Catalan tweets. Bachelor thesis, Universitat de Lleida.

[87] Gómez-Adorno, H., Markov, I., Baptista, J., Sidorov, G., and Pinto, D. (2017). Discriminating between Similar Languages Using a Combination of Typed and Untyped Character N-grams and Words. In Proceedings of the Fourth Workshop on NLP for Similar Languages, Varieties and Dialects, Valencia, Spain, pp. 137-145.

[88] Alrifai, K., Rebdawi, G., and Ghneim, N. (2017). Arabic Tweeps Gender and Dialect Prediction – Notebook for PAN at CLEF 2017. In Linda Cappellato, Nicola Ferro, Lorraine Goeuriot, and Thomas Mandl, editors, Working Notes Papers of CLEF 2017 Evaluation Labs and Workshop, Dublin, Ireland. CEUR-WS.org. URL: http://ceur-ws.org/Vol-1866/.

[89] Malmasi, S., and Dras, M. (2017). Feature Hashing for Language and Dialect Identification. In Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Short Papers), Vancouver, Canada, pp. 399-403.

[90] Miura, Y., Taniguchi, T., Taniguchi, M., SMisawa, S., and Ohkuma, T. (2017). Using Social Networks to Improve Language Variety Identification with Neural Networks. In Proceedings of the 8th International Joint Conference on Natural Language Processing, Taipei, Taiwan, pp. 263-270.

[91] Tellez, E.S., Miranda-Jiménez, S., Graff, M., and Moctezuma, D. (2017). Gender and Language-variety Identification with MicroTC – Notebook for PAN at CLEF 2017. In Linda Cappellato, Nicola Ferro, Lorraine Goeuriot, and Thomas Mandl, editors, Working Notes Papers of CLEF 2017 Evaluation Labs and Workshop, Dublin, Ireland. CEUR-WS.org. URL: http://ceur-ws.org/Vol-1866/.

[92] Prager, J.M. (1999). Linguini: Language Identification for Multilingual Documents. Journal of Management Information Systems, 1999, S. 1–11.

[93] Bilcu, E.B. and Astola, J. (2006). A Hybrid Neural Network for Language Identification from Text. In Proceedings of the 16th IEEE Workshop on Machine Learning for Signal Processing, Signal Processing Society, Maynooth, Ireland, pp. 253-258.

[94] Murthy, K.N. and Kumar, G.B. (2006). Language Identification from Small Text Samples. Journal of Quantitative Linguistics, 13(1):57-80.

[95] Babu, J. V., and Baskaran, S. (2005). Automatic Language Identification Using Multivariate Analysis. In Proceedings of the 6th International Conference on Intelligent Text Processing and Computational Linguistics, Springer, Berlin, Heidelberg, pp. 789-792.

[96] Hayati, K. (2004). Language Identification on the World Wide Web. Master's thesis, University of California Santa Cruz, Santa Cruz, California, USA.

[97] Malmasi, S., Refaee, E., and Dras, M. (2015). Arabic Dialect Identification using a Parallel Multidialectal Corpus. In Proceedings of the 14th Conference of the Pacific Association for Computational Linguistics, PACLING'15, Bali, Indonesia, pp. 209-217.

[98] Cazamias, J., Dixit, C., and Marek, M. (2015). Large-Scale Language Classification - Writing a Detector for 200 Languages on Twitter. Stanford course report.

[99] Hochreiter, S. and Schmidhuber, J. (1997). Long short-term memory. Neural computation, 9(8):1735–1780.

[100] Samih, Y., Maharjan, S., Attia, M., Kallmeyer, L., and Solorio, T. (2016). Multilingual Code-switching Identification via LSTM Recurrent Neural Networks. In: Proceedings of the Second Workshop on Computational Approaches to Code Switching, Austin, TX, USA, pp. 50-59.

[101] Bjerva, J. (2016). Byte-based Language Identification with Deep Convolutional Networks. In Proceedings of the Third Workshop on NLP for Similar Languages, Varieties and Dialects, Osaka, Japan, pp. 119-126.

[102] Jurgens, D., Tsvetkov, Y., and Jurafsky, D. (2017). Incorporating Dialectal Variability for Socially Equitable Language Identification. In: Proceedings of the 55th Annual Meeting

of the Association for Computational Linguistics (Short Papers), Vancouver, Canada, 2:51-57.

[103] Kocmi, T., and Bojar, O. (2017). LanideNN: Multilingual Language Identification on Character Window. In Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Long Papers, Valencia, Spain, 1:927-936.

[104] Jaech, A., Mulcaire, G., Hathi, S., Ostendorf, M., and Smith, N.A. (2016a). Hierarchical Character-Word Models for Language Identification. In Proceedings of the Fourth International Workshop on Natural Language Processing for Social Media, Austin, TX, USA, pp. 84-93.

[105] Jaech, A., Mulcaire, G., Hathi, S., Ostendorf, M., and Smith, N.A. (2016b). Neural Model for Language Identification in Code-Switched Tweets. In Proceedings of the Second Workshop on Computational Approaches to Code Switching, Austin, TX, USA, pp. 60-64.

[106] Li, Y., Cohn, T., and Baldwin, T. (2018). What's in a domain? learning domain robust text representations using adversarial training. In Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics-Human Language Technologies (NAACL HLT 2018), New Orleans, USA, pp. 474-479.

[107] Adouane, W. and Dobnik, S. (2017). Identification of Languages in Algerian Arabic Multilingual Documents. In Proceedings of The Third Arabic Natural Language Processing Workshop (WANLP 2017), Valencia, Spain, pp. 1-8.

[108] Guzmàn, G.A., Ricard, J., Serigos, J., Bullock, B., and Toribio, A.J. (2017). Moving Code-switching Research Toward More Empirically Grounded Methods. In Thierry Declerck and Sandra K¨ubler, editors, Proceedings of the Workshop on Corpora in the Digital Humanities (CDH 2017), Bloomington, USA, pp. 1-9.

[109] Rijhwani, S., Sequeira, R., Choudhury, M., Bali, K., and Maddila, C.S. (2017). Estimating Code-Switching on Twitter with a Novel Generalized Word-Level Language Detection Technique. In Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, Vancouver, Canada, pp. 1971-1982.

[110] Samih, A. (2017). Dialectal Arabic Processing Using Deep Learning. PhD thesis, Heinrich-Heine-Universit¨at D¨usseldorf, D¨usseldorf, Germany.

[111] Darwish, K., Sajjad, H., and Mubarak, H. (2014). Verifiably Effective Arabic Dialect Identification. In Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP 2014), Doha, Qatar, pp. 1465-1468.

[112] Takçı, H., and Güngör, T. (2012). A High Performance Centroid-based Classification Approach for Language Identification. Pattern Recognition Letters, 33(16), 2077–2084.

[113] Ng, C.-C. and Selamat, A. (2009). Improved Letter Weighting Feature Selection on Arabic Script Language Identification. In Proceedings of the 1st Asian Conference on Intelligent Information and Database Systems (ACIIDS 2009), Dong Hoi, Vietnam. IEEE, pp. 150–154.

[114] Kerwin, T. (2006). Classification of Natural Language Based on Character Frequency. Ohio Supercomputer Center.

[115] Basile, A., Dwyer, G., Medvedeva, M., Rawee, J., Haagsma, H., and Nissim, M. (2017). NGrAM: New Groningen Author-profiling Model-Notebook for PAN at CLEF 2017. In: Working Notes Papers of CLEF 2017 Evaluation Labs and Workshop, Dublin, Ireland.

[116] Bestgen, Y. (2017). Improving the Character N-gram Model for the DSL Task with BM25 Weighting and Less Frequently Used Feature Sets. In Proceedings of the Fourth Workshop on NLP for Similar Languages, Varieties and Dialects (VarDial), pp. 115–123, Valencia, Spain.

[117] Simaki, V., Simakis, P., Paradis, C., and Kerren, A. (2017). Identifying the Authors' National Variety of English in Social Media Texts. In Proceedings of the International Conference Recent Advances in Natural Language Processing (RANLP 2017), pp. 671–678, Varna, Bulgaria. INCOMA Ltd.

[118] DIBCO, 2009. http://www.iit.demokritos.gr/ bgat/ DIBCO2009/benchmark/

[119] DIBCO, 2013. http://utopia.duth.gr/ ipratika/DIBCO2013/ benchmark/

[120] DIBCO, 2017. https://vc.ee.duth.gr/dibco2017/benchmark/

[121] Xie, J., Jiang, S., Xie, W., and Gao, X. (2011). An Efficient Global K-means Clustering Algorithm. Journal of Computers, 6(2):271-279.

[122] Kanungo, T., Mount, D.M., Netanyahu, N.S., Piatko, C.D., Silverman, R., and Wu, A.Y. (2002). An Efficient k-means Clustering Algorithm: Analysis and Implementation. IEEE Transactions on Pattern Analysis and Machine Intelligence. 24(7), 881-892.

[123] Kingma, D. and Ba, J. (2015). Adam: A method for stochastic optimization. In Proceedings of International Conference for Learning Representations 2015, San Diego, USA, http://arxiv.org/abs/1412.6980.

[124] DIBCO, 2011. http://utopia.duth.gr/ ipratika/ DIBCO2011/benchmark/ dataset/

[125] H-DIBCO, 2010. http://users.iit.demokritos.gr/ bgat/H-DIBCO2010/benchmark/

[126] H-DIBCO, 2012. http://utopia.duth.gr/ ipratika/ HDIBCO2012/benchmark/dataset/

[127] H-DIBCO, 2014. http://users.iit.demokritos.gr/ bgat/ HDIBCO2014/benchmark

[128] H-DIBCO, 2016. https://vc.ee.duth.gr/h-dibco2016/benchmark/

[129] H-DIBCO, 2018. http://vc.ee.duth.gr/h-dibco2018/benchmark/

[130] Sehad, A., Chibani, Y., Cheriet, M., and Yaddaden, Y. (2013). Ancient degraded document image binarization based on texture features. IEEE 8th International Symposium on Image and Signal Processing and Analysis (ISPA), Trieste, Italy, pp 189-193. Doi: 10.1109/ISPA.2013.6703737.

[131] Saddami, K., Munadi, K., Muchallil, S., and Arnia, F. (2017). Improved Thresholding Method for Enhancing Jawi Binarization Performance. IEEE 14th IAPR International Conference on Document Analysis and Recognition (ICDAR), Kyoto, Japan, pp 1108-1113. Doi: 10.1109/ICDAR.2017.183

[132] Pratikakis, I., Gatos, B., Ntirogiannis, K. (2010). H-DIBCO 2010 Handwritten document image binarization competition. Proc. 12th Int. Conf. Frontiers in Handwriting Recognition, IEEE. pp. 727-732.

[133] Trier, O.D. and Taxt, T. (1995). Evaluation of binarization methods for document images. IEEE Transaction on Pattern Analysis and Machine Intelligence, 17(3):312-315.

[134] Lu H., Kot, A.C., and Shi, Y.Q. (2004). Distance-reciprocal distortion measure for binary document images. IEEE Signal Processing Letters, 11(2): 228-231. Doi: 10.1109/LSP.2003.821748

[135] Pratikakis, I., Zagoris, K., Barlas, G., and Gatos, B. (2016). ICFHR2016 handwritten document image binarization contest (H-DIBCO 2016). Proc. 15th Int. Conf. on Frontiers in Handwriting Recognition, IEEE, pp. 619-623.

[136] Bhowmik, S., et al. (2019). GiB: A Game Theory Inspired Binarization Technique for Degraded Document Images. IEEE Transactions on Image Processing, 28(3):1443-1455.

[137] Su, B., Lu, S., and Tan, C.L. (2010). Binarization of historical handwritten document images using local maximum and minimum filter. Proc. 9th IAPR Int. Workshop on Document Analysis Systems, Boston, Massachusetts, USA. pp. 159-166.

[138] Badekas, E., and Papamarkos, N. (2007). Optimal combination of document binarization techniques using a self-organizing map neural network. Eng. Appl. Artif. Intell., 20(1):11-24.

[139] Roy, P.P., Rayar, F., Ramel, J.Y. (2015). Word spotting in historical documents using primitive code book and dynamic programming. Image Vis. Comput., 44:15-28.

[140] Lu, S., Li, L., Tan, C.L., Member, S. (2008). Document Image Retrieval through Word Shape Coding. IEEE Transactions on Pattern Analysis and Machine Intelligence, 30(11):1913-1918.

[141] Rath, T.M., Manmatha, R. (2003). Word image matching using dynamic time warping. IEEE Computer Society Conference on Computer Vision and Pattern Recognition 2, II-II.

[142] Dutta, K., Krishnan, P., Mathew, M., Jawahar, C.V. (2018a). Towards Spotting and Recognition of Handwritten Words in Indic Scripts. 16th International Conference on Frontiers in Handwriting Recognition (ICFHR), IEEE.

[143] Roy, P.P., Bhunia, A.K., Bhattacharyya, A., and Pal, U. (2019). Word searching in scene image and video frame in multi-script scenario using dynamic shape coding. Multimedia Tools and Applications, 78:7767–7801. Doi: https://doi.org/10.1007/s11042-018-6484-5.

[144] Khurshid, K., Faure, C., Vincent, N. (2012). Word spotting in historical printed documents using shape and sequence comparisons. Pattern Recognition, 45(7):2598-2609.

[145] Papandreou, A., Gatos, B., Louloudis, G. (2014). An adaptive zoning technique for efficient word retrieval using dynamic time warping. Proceedings of the First International Conference on Digital Access to Textual Cultural Heritage (DATeCH), pp. 147-152.

[146] Fernández-Mota, D., Lladós, J., and Fornés, A. (2014). A graph-based approach for segmenting touching lines in historical handwritten documents. IJDAR, 17:293-312.

[147] Louloudis, G., Gatos, B., Pratikakis, I., and Halatsis, C. (2009). Text line And Word Segmentation of Handwritten Documents. Pattern Recognition, 42(12):3169-3183.

[148] Huang, C. and Srihari, S.N. (2008). Word Segmentation of Off-line Handwritten Documents. Proceedings of SPIE - The International Society for Optical Engineering, 6815:68150.

[149] Papavassiliou, V., Stafylakis, T., Katsouros, V., and Carayannis, G. (2010). Handwritten document image segmentation into text lines and words. Pattern Recognition, 43:369-377.

[150] Chen. Y. and Gupta. M.R. (2010). EM demystified: An Expectation-Maximization Tutorial. Technical Report, Department of Electrical Engineering, University of Washington.

[151] Mahadevan, U. and Nagabushnam, R.C. (1995). Gap Metrics for Word Separation in Handwritten Lines. Proceedings of 3rd ICDAR.

[152] Shokoohi-Yekta, M., Hu, B., Jin, H., Wang, J., and Keogh, E. (2017). Generalizing Dynamic Time Warping to the Multi-Dimensional Case Requires an Adaptive Approach. Data Min Knowl Discov, 31(1):1–31.

[153] Gowayyed, M.A., Torki, M., Hussein, M.E., El-Saban, M. (2013). Histogram of Oriented Displacements (HOD): Describing Trajectories of Human Joints for Action Recognition. Proceedings of the Twenty-Third International Joint Conference on Artificial Intelligence, pp. 1351-1357.

[154] Keras documentation: Getting started with the Keras Sequential model, https://keras.io/getting-started/sequential-model-guide/. Accessed 20 June. 2022.

[155] Gatos, B., Stamatopoulos, N., and Louloudis, G. (2011). ICDAR2009 handwriting segmentation contest. Int J. Doc. Anal Recognit (IJDAR), 14(1):25-33.

[156] Gatos, B., Stamatopoulos. N., and Louloudis, G. (2009). ICDAR2009 handwriting segmentation contest. Proceedings of ICDAR pp. 1393-1397.

[157] Stamatopoulos, N., Gatos, B., Louloudis, G., Pal, U., and Alaei, A. (2013). ICDAR 2013 handwriting segmentation contest. Proceedings of 12th ICDAR pp. 1402–1406.

[158] Chaudhuri, B.B. and Pal, U. (1997). An OCR system to read two Indian language scripts: Bangla and Devnagari (Hindi). Proceedings of the fourth international conference on document analysis and recognition, pp. 1011–1015.

[159] Karmakar, P., Nayak, B., and Bhoi, N. (2014). Line and Word segmentation of a printed text document. Int J. Comput. Sci. Inf. Technol., 5(1):157-160.

[160] Jain, S. and Singh, H. (2014). A novel approach for word segmentation in correlation based OCR system. Int J Comput Appl., 99(18):12–20.

[161] Sharma, M.K. and Dhaka, V.P. (2016). Pixel plot and trace based segmentation method for bilingual handwritten scripts using feedforward neural network. Neural Comput. Appl., 27(7):1817-1829.

[162] Dahake, D., Sharma, R.K., and Singh, H. (2017). On segmentation of words from online handwritten Gurmukhi sentences. Proceedings of 2017 2nd international conference on man and machine interfacing (MAMI).

[163] Sharma, M.N. and Dhaka, V.S. (2020). Segmentation of handwritten words using structured support vector machine. Pattern Analysis and Applications, 23:1355-1367.

[164] Konidaris, T., et al. (2007). Keyword-guided word spotting in historical printed documents using synthetic data and user feedback. Int. J. Document Anal. Recognit., 9(2-4):167-177.

[165] Jindal, P. and Jindal, B. (2015). Line and word segmentation of handwritten text documents written in Gurmukhi script using mid-point detection technique. International Journal of Advance Research in Science and Engineering 4(1):11-19.

[166] Schomaker, L. and Vuurpijl, L. (2000). Forensic Writer Identification: A Benchmark Data Set and a Comparison of Two Systems, Technical Report, NICI, Nijmegen.

[167] Dutta, K., Krishnan, P., Mathew, M., Jawahar, C.V. (2018b). Offline Handwriting Recognition on Devanagari using a new Benchmark Dataset. 13th IAPR International Workshop on Document Analysis Systems (DAS), IEEE, pp. 25-30.

[168] Simard, P.Y., Steinkraus, D., Platt, J.C. (2003). Best practices for convolutional neural networks applied to visual document analysis. 7th International Conference on Document Analysis and Recognition, Edinburgh, UK. 10.1109/ICDAR.2003.1227801.

[169] Chatzichristofis, S.A., Zagoris, K., Arampatzis, A. (2011). The trec files: the (ground) truth is out there. 34th international ACM SIGIR conference on Research and development in Information, ser. SIGIR '11. New York, NY, USA: ACM, pp. 1289-1290.

[170] Krishnan, P., Dutta, K., Jawahar, C.V. (2018). Word spotting and recognition using deep embedding. 13th IAPR International Workshop on Document Analysis Systems (DAS), IEEE.

[171] Jauhiainen, T., Lui, M., Zampieri, M., Baldwin, T., and Lindén, K. (2019). Automatic Language Identification in Texts: A Survey. Journal of Artificial Intelligence Research 65: 675-782.

[172] Bird, S., Klein, E., and Loper, E. (2009). Natural Language Processing with Python. O'Reilly Media.

[173] Newman, P. (1987). Foreign language identification – a first step in translation. In: Proceedings of the 28th Annual Conference of the American Translators Association, pp. 509-516.

[174] Cho, K., Merrienboer, B. V., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., and Yoshua Bengio, Y. (2014). Learning phrase representations using rnn encoder–decoder for statistical machine translation. In Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), Doha, Qatar, pp, 1724-1734.

[175] Gers, F.A., Schraudolph, N.N, and Schmidhuber, J. (2003). Learning precise timing with LSTM recurrent networks. Journal of Machine Learning Research, 3:115-143.

[176] Gers, F.A., Schmidhuber, J., and Cummins, F. (2000). Learning to forget: Continual prediction with LSTM. Neural Computation, 12(10):2451-2471.

[177] Graves, A., Liwicki, M., Fernández, S., Bertolami, R., Bunke, H., and Ju¨rgen Schmidhuber, J. (2009). A Novel Connectionist System for Unconstrained Handwriting Recognition. IEEE Transactions on Pattern Analysis and Machine Intelligence, 31(5):855-897. Doi: 10.1109/TPAMI.2008.137.

[178] Graves, A. and Schmidhuber, J. (2005). Framewise phoneme classification with bidirectional LSTM and other neural network architectures, 18(5-6):602-610.

[179] Poomka, P., Pongsena, W., Kerdprasop, N., and Kerdprasop, K. (2019). SMS Spam Detection Based on Long Short-Term Memory and Gated Recurrent Unit. International Journal of Future Computer and Communication, 8(1):11-15.

[180] Pennington, J., Socher, R., and Manning, C. (2014). Glove: Global vectors for word representation. In Proc. of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), pp. 1532-1543. Doi: 10.3115/v1/D14-1162.Doi: 10.3115/v1/D14-1162.

[181] Mikolov, T., Chen, K., Corrado, G., Dean, J. (2013). Efficient Estimation of Word Representations in Vector Space. Doi: https://arxiv.org/abs/1301.3781

[182] Tomas, M., Ilya, S., Kai, C., Corrado, G.S., and Jeff, D. (2013). Distributed representations of words and phrases and their compositionality. Advances in Neural Information Processing Systems. arXiv:1310.4546.

[183] Universal Declaration of Human Rights (UDHR) Translation Project, http://www.ohchr.org/EN/UDHR/Pages/Introduction.aspx, Accessed June 20, 2022.

[184] Universal Declaration of Human Rights (UDHR) corpus, http://research.ics.aalto.fi/cog/data/udhr/, Accessed June 20, 2022.

[185] Vatanen, T., Väyrynen, J.J, and Virpioja, S. (2010) Language identification of short text segments with n-gram models. In Proceedings of the Seventh International Conference on Language Resources and Evaluation (LREC'10). European Language Resources Association (ELRA), pp. 3423-3430.

[186] He S, Schomaker S (2019) Deep adaptive learning for writer identification based on single handwritten word images. Pattern Recognit. 88:64-74.

[187] Indolia, S., Goswami, A.K., Mishra, S.P., and Asopa, P. (2018). Conceptual Understanding of Convolutional Neural Network- A Deep Learning Approach. Procedia Computer Science 132:679–688, doi: 10.1016/j.procs.2018.05.069.

[188] Albawi, S., Mohammed, T.A., Al-Zawi, S. (2017). Understanding of a convolutional neural network. In: IEEE International Conference on Engineering and Technology, pp. 1-6.

[189] Chen, S., Wang, Y., Lin, C-T, Ding, W., Cao, Z. (2019). Semi-supervised feature learning for improving writer identification. Information Sciences, 482:156–170, doi: https://doi.org/10.1016/j.ins.2019.01.024.

[190] Zhang, Z. and Sabuncu, M. (2018). Generalized Cross Entropy Loss for Training Deep Neural Networks with Noisy Labels. axXiv preprint, arXiv: 1805.07836v4.

[191] .Guo, G., Fu, Y., Dyer, C.R., and T. S. Huang, T.S. (2008). Image-based human age estimation by manifold learning and locally adjusted robust regression. IEEE Transactions on Image Processing, 17(7):1178-1188.

[192] Geng, X., Zhou, Z.-H., and Smith-Miles, K. (2007). Automatic age estimation based on facial aging patterns. IEEE Transactions on Pattern Analysis and Machine Intelligence, 29(12):2234-2240.

[193] He, S. and Schmaker, L. (2014). Delta-n hinge: rotation invariant features for writer identification. Proceedings of 22nd IEEE International Conference on Pattern Recognition, pp. 2023-2028, doi: 10.1109/ICPR.2014.353.

[194] Srihari, S.N., Cha, S-H., Arora, H., and Lee, S. (2002). Individuality of handwriting. J. Forensic Sci., 47(4):856-872.

[195] Fiel, S. and Sablatnig, R. (2012). Writer Retrieval and Writer Identification Using Local Features. 10th IAPR Int. Work. Doc. Anal. Syst., pp. 145-149.

[196] Dargan, S. and Kumar, M. (2019). Writer identification system for indic and non-indic scripts: State-of-the-art survey. Arch. Comput. Methods Eng., 26(4):1283-1311.

[197] Rehman, A., Naz, S., and Razzak, M.I. (2019). Writer identification using machine learning approaches: a comprehensive review. Multimedia Tools and Applications, 78:10889-10931.

[198] Adak, C., Chaudhuri, B.B., and Blumenstein, M. (2019). An Empirical Study on Writer Identification and Verification from Intra-Variable Individual Handwriting. IEEE Access, 7:24738-24758.

[199] Esuli, A., Moreo, A., and Sebastiani, F. (2019). Funnelling: A New Ensemble Method for Heterogeneous Transfer Learning and its Application to Cross-Lingual Text Classification. ACM Transactions on Information Systems, 1(1):1-29.

[200] Sanasam, I., Choudhary, P., and Singh, K.M. (2020). Line and word segmentation of handwritten text document by mid-point detection and gap trailing. Multimedia Tools and Applications, 79:30135-30150. https://doi.org/10.1007/s11042-020-09416-1.

[201] Omayio, E.O., Indu, S., and Panda, J. (2022b). Word Segmentation by Component Tracing and Association (CTA) Technique. Journal of Engineering Research. Doi: https://doi.org/10.36909/jer.15207

[202] Schomaker, L. and Bulacu, M. (2004). Automatic writer identification using connected-component contours and edge-based features of uppercase western script, IEEE Transactions on Pattern Analysis and Machine Intelligence, 26(6):787-798.

[203] Arazi, B. (1977). Handwriting identification by means of runlength measurements. IEEE Trans. Syst., Man and Cybernetics, 7(12):878-881.

[204] Press, W., Teukolsky, S., Vetterling, W., and Flannery, B. (1992). Numerical Recipes in C: The Art of Scientific Computing, second ed. Cambridge Univ. Press.

[205] Ting, K.M. and Witten, I.W. (1999). Issues in stacked generalization. Journal of Artificial Intelligence Research, 10:271-289.

[206] Wolpert, D. (1992). Stacked generalizations. Neural networks, 5(2):241-259.

[207] Marti, U.V. and Bunke, H. (2002). The IAM-database: An english sentence database for offline handwriting recognition. Int. J. Document Anal. Recognit., 5(1):39-46.

[208] Wu, X., Tang, Y., and Bu, W. (2014). Offline text-independent writer identification based on scale invariant feature transform, IEEE Trans. Inf. Forensics Secur., 9 (3):526-536.

[209] Ghiasi, G. and Safabakhsh, R. (2013). Offline text-independent writer identification using codebook and efficient code extraction methods. Image Vis. Comput., 31(5): 379-391.

[210] Khalifa, E., Al-Maadeed, S., Tahir, M.A., Bouridane, A., and Jamshed, A. (2015). Offline writer identification using an ensemble of grapheme codebook features. Pattern Recogn. Lett., 59:18-25.

[211] Siddiqi, I. and Vincent, N. (2010). Text independent writer recognition using redundant writing patterns with contour-based orientation and curvature features. Pattern Recognit., 43(11):3853-3865.

[212] Tang, Y., Wu, X., and Bu, W. (2013). Offline Text-independent Writer Identification Using Stroke Fragment and Contour Based Features. IEEE IEEE International Conference on Biometrics (ICB), doi: 10.1109/ICB.2013.6612988.