HANDWRITTEN MALAYALAM WORD RECOGNITION USING HYBRID DEEP NEURAL NETWORK MODELS

A Thesis Submitted to
UNIVERSITY OF CALICUT
in partial fulfilment of the requirements for the award of the degree of

DOCTOR OF PHILOSOPHY IN COMPUTER SCIENCE
Under the Faculty of Science

By

ANJU A T

Under the guidance of

Dr. BINU P. CHACKO
Associate Professor, Department of Computer Science,
Prajyoti Niketan College, Pudukkad (Guide)

Prof. (Dr.) MOHAMED BASHEER K.P
Principal and Professor,
Amal College of Advanced Studies, Nilambur (Co-Guide)

P.G & RESEARCH DEPARTMENT OF COMPUTER SCIENCE
Sullamussalam Science College, Areekode - 673639
(Affiliated to the University of Calicut)
Malappuram Dist., Keralam, India
January 2026

DECLARATION

I, Anju A T, hereby declare that the work presented in the thesis entitled "Handwritten Malayalam Word Recognition using Hybrid Deep Neural Network Models" is based on the original work done by me under the guidance of Dr. Binu P. Chacko (Guide) and Prof. (Dr.) Mohamed Basheer K. P. (Co-Guide) in the PG & Research Department of Computer Science, Sullamussalam Science College, Areekode, Kerala, and has not been included in any other thesis submitted previously for the award of any degree. The contents of the thesis have undergone a plagiarism check using iThenticate software at the C.H.M.K. Library, University of Calicut, and the similarity index is within the permissible limit. I also declare that the thesis is free from AI-generated content.

Anju A. T

Areekode
January, 2026

Dr. Binu P. Chacko
Research Guide
PG & Research Department of Computer Science, Sullamussalam Science College, Areekode, Kerala, India

Prof. (Dr.) Mohamed Basheer K.P
Research Co-Guide
PG & Research Department of Computer Science, Sullamussalam Science College, Areekode
Kerala, India

SULLAMUSSALAM SCIENCE COLLEGE
AREEKODE

SULLAMUSSALAM SCIENCE COLLEGE
AREEKODE

Phone :0483-2850700
Mob :+919895811093
Website : www.sscollege.ac.in
E-mail : mail@sscollege.ac.in

Aided College | Affiliated to the University of Calicut | Re-accredited by NAAC at A Grade

Ref.

Date :.....

This is to certify that the thesis entitled "HANDWRITTEN MALAYALAM WORD RECOGNITION USING HYBRID DEEP NEURAL NETWORK MODELS", submitted to the University of Calicut, for partial fulfillment of the requirements for the award of the degree of Doctor of Philosophy (Ph.D.) in Computer Science, is a bonafide research work done by Mrs. Anju A T under our supervision and guidance in the PG & Research Department of Computer Science, Sullamussalam Science College, Areekode, Malappuram, Kerala. The content embodied in this thesis, in full or in parts, has not been submitted to any other University or Institute for the award of any degree.

The thesis is revised as per the modifications and recommendations reported by the adjudicators. Soft copy attached is the same as that of the revised copy. The thesis is submitted as such to the University of Calicut with reference to the letter number No. 9358/RESEARCH-C-ASST-1/2026/Admn dated 11-05-2026

Dr. Binu P. Chacko
Research Guide
PG & Research Department of Computer Science,
Sullamussalam Science College,
Areekode, Kerala, India

Prof. (Dr.) Mohamed Basheer K.P
Research Co-Guide
PG & Research Department of Computer Science,
Sullamussalam Science College,
Areekode, Kerala, India

Areekode
May 2026

SULLAMUSSALAM SCIENCE COLLEGE * AREEKODE
UGRAPURAM P.O
MALAPPURAM DT.
PIN-673639

UNIVERSITY OF CALICUT

CERTIFICATE ON PLAGIARISM CHECK

1.	Name of the Research Scholar	ANJU A T	
2.	Title of thesis / dissertation	HANDWRITTEN MALAYALAM WORD RECOGNITION USING HYBRID DEEP NEURAL NETWORK MODELS	
3.	Name of the Supervisor	Dr. BINU P. CHACKO Prof. (Dr.) MOHAMED BASHEER K. P (Co-guide)	
4.	Department/Institution	PG & RESEARCH DEPARTMENT OF COMPUTER SCIENCE,SULLAMUSSALAM SCIENCE COLLEGE, AREEKODE	
5.	Similar content (%) identified	Non Core	Core
		Introduction/ Theoretical overview/Review of literature/ Materials & Methods/ Methodology	**Analysis/Result/Discussion / Summary/Conclusion/ Recommendations**
		1	**2**
	Acceptable maximum limit (%)	10	10
6.	Software used	iThenticate	
7.	Date of verification	01.01.2026	

**Report on plagiarism check, specifying included/excluded items with % of similarity to be attached.*

Dr. Nasirudheen. T
Assistant Librarian
University of Calicut, Kerala.

Checked by (with name , designation & signature)

Name and signature of the Researcher ANJU A.T

Name and signature of the Supervisor.

Binu P Chacko

The Doctoral Committee* has verified the report on plagiarism check with the contents of the thesis, as summarized above and appropriate measures have been taken to ensure originality of the Research accomplished herein.

Dr. Mustafa Farook.P

Name & Signature of the HoD/HoI (Chairperson of the Doctoral Committee)

PRINCIPAL
SULLAMUSSALAM SCIENCE COLLEGE
AREEKODE, UGRAPURAM (P.O)
MALAPPURAM (Dt), PIN-673639

**In case of languages like Malayalam, Tamil etc..on which no software is available for plagiarism check, a manual check shall be made by the Doctoral Committee, for which an additional certificate has to be attached.*

SULLAMUSSALAM SCIENCE COLLEGE * AREEKODE * UGRAPURAM P.O MALAPPURAM DT. PIN-673639

ACKNOWLEDGEMENT

The successful completion of this research work would not have been possible without the guidance, encouragement, and support of many individuals who contributed in various ways throughout my Ph.D. journey.

I express my most profound sense of gratitude to my research guide, Dr. Binu P. Chacko, Research Guide, P.G. and Research Department of Computer Science, Sullamussalam Science College, Areekode, for his constant encouragement, motivation, and unwavering support throughout this research. His timely assistance, detailed guidance, and availability at every stage of the work were instrumental in shaping this study. He consistently provided clear direction, valuable suggestions, and constructive feedback without delay, which greatly helped me overcome research challenges and maintain steady progress. His dedication and commitment as a mentor have played a significant role in the successful completion of this work.

I sincerely acknowledge my research co-guide, Prof. (Dr.) Mohamed Basheer K. P., Research Guide, P.G. and Research Department of Computer Science, Sullamussalam Science College, Areekode, for his continuous support, encouragement, and motivation throughout my research journey. He stood by me during difficult phases, offering guidance and reassurance whenever needed. Beyond academic support, his mentoring helped me grow personally, teaching me values that shaped me not only as a researcher but also as a better individual.

I extend my heartfelt gratitude to Dr. Musthafa Farook P., Principal, Sullamussalam Science College, Areekode, for providing a conducive academic environment and institutional support necessary for carrying out this research. I also sincerely thank the former Principal, Dr. Muhamed Ilyas P., who has recently retired, for his constant encouragement, guidance, and support during all stages of my research work. His confidence in me served as a great source of motivation.

I express my sincere thanks to the members of my Research Advisory Committee for their invaluable suggestions and constructive feedback, which significantly improved this research. I am grateful to Dr. Lajish V. L., Associate Professor, Department of Computer Science, University of Calicut; Dr. Shameem Kappan, Head, P.G. and Research Department of Computer Science, Sullamussalam Science

College, Areekode; and Dr. Vasudevan T. M., Professor and Head, Department of Library and Information Science, University of Calicut, for their guidance and encouragement.

I would also like to express my sincere gratitude to Dr. Vivek P., Director, IT Centre, Kannur University, for his valuable support and encouragement during the early stage of this research work. His motivation and guidance provided a strong foundation for carrying out this study.

I also extend my heartfelt thanks to Dr. Shailesh Sivan., Assistant Professor, Cochin University of Science and Technology, for his continuous support, insightful suggestions, and encouragement throughout my research journey. His valuable guidance and timely advice greatly contributed to the successful completion of this work.

I sincerely express my heartfelt gratitude to Dr. Haulath K., Assistant Professor at EMEA College, for her exceptional support, constant encouragement, and detailed guidance throughout the thesis submission process. Her timely assistance and valuable guidance at every stage played a vital role in the successful completion and submission of this thesis.

I extend my sincere appreciation to all the teaching and non-teaching staff of Sullamussalam Science College, Areekode, for their constant cooperation, administrative assistance, and moral support throughout my research period. I would especially like to acknowledge Mr. M. Abdussalam., Superintendent, for his wholehearted support in handling academic and official matters, which greatly facilitated the smooth progress of my research.

I am deeply indebted to my family for their unconditional love, patience, and moral support, which remained a constant source of strength throughout this journey. I also thank my friends and all well-wishers for their encouragement and support.

Anju A T

ABSTRACT

Handwritten document analysis remains a significant challenge in pattern recognition, particularly for scripts with complex morphology such as Malayalam. Even today, handwritten Malayalam documents are widely used in government, police, and legal settings, yet there are no reliable automated methods for reading or retrieving information from them. The lack of publicly available handwritten Malayalam word datasets further restricts progress, limiting the use of deep learning models. This study addresses these challenges through a comprehensive research effort involving the construction of a new handwritten word dataset, the development of a hybrid deep learning–based handwritten word recognition framework, and the design of a practical document retrieval system based on recognized word content.

The first component of this work focuses on constructing the Malayalam Handwritten Word Dataset (MHWD). Handwritten material was sourced from police First Information Statements (FIS) and further enriched with samples provided by writers from different age groups. All document images underwent a uniform preprocessing sequence that included input image enhancement, binarization, contour detection, and word-level segmentation using bounding boxes. To define the vocabulary, the most frequently occurring words and domain specific words in the collected FIS documents were manually shortlisted, yielding 321 word categories. The number of samples from real documents was insufficient and imbalanced. So, additional handwritten instances were collected from individuals aged 10 to 65, yielding 150 samples per class. This balanced dataset serves as a dependable base for training and evaluating Malayalam handwritten word recognition methods.

The second component of this research introduces the proposed handwritten Malayalam word recognition model, termed TrA-MHWR. The model integrates a DeiT-based vision transformer for feature extraction with an Attention-based Feedforward Neural Network (AFFNN) as the recognition module. The transformer

encoder provides a rich, context-aware representation of word images, while the AFFNN enhances the discriminative capability of these features through multi-head attention and residual learning. This combination allows the model to effectively handle handwriting variability and visually similar word forms commonly encountered in real-world documents.

The third contribution of the study is the development of a practical document retrieval system built upon the outputs of the word recognition model. For this application-oriented stage, a lightweight EfficientNetB0-based recognition model was adopted due to its low parameter count. The same preprocessing and segmentation pipeline was applied to each page of the collected handwritten documents. The EfficientNet recognizer was used to identify words belonging to the 321-word lexicon, enabling partial textual representation of each document. The recognized subset of words provides sufficient information for effective retrieval. A TF–IDF (Term Frequency-Inverse Document Frequency) representation was constructed from the recognized words, and Latent Dirichlet Allocation (LDA) was used to model semantic relationships between documents. Each document is ranked according to its similarity to the query, and the top-5 documents are returned as the retrieval output.

Overall, this thesis presents a complete framework, from dataset construction and recognition model development to real-world document retrieval. The developed methodologies provide a foundation for further research on low-resource scripts and open the way for scalable deployment in government, legal, and administrative contexts where handwritten records continue to play a significant role.

Keywords: Handwritten Malayalam Word Recognition, Vision Transformer, Document Image Analysis, Word Segmentation, Low-Resource Scripts, Handwritten Word Dataset, Document Retrieval, Attention Mechanism.

സംഗ്രഹം

കൈയ്യെഴുത്ത് ഡോക്യുമെന്റുകളുടെ വിശകലനം പാറ്റേൺ തിരിച്ചറിയലിൽ ഒരു പ്രധാന വെല്ലുവിളിയാണ്. പ്രത്യേകിച്ചും മലയാളം പോലെ സങ്കീർണമായ ലിപികളുടെ കാര്യത്തിൽ. ഇന്നും മലയാളത്തിലുള്ള കയ്യെഴുത്ത രേഖകൾ ഗവൺമെൻറ്, പോലീസ്, നിയമ മേഖലകളിൽ വ്യാപകമായി ഉപയോഗിക്കുന്നുണ്ടെങ്കിലും അവയിൽ നിന്ന് വിവരങ്ങൾ വീണ്ടെടുക്കാൻ ആശ്രയിക്കാവുന്ന തന്നത്താൻ പ്രവർത്തിക്കുന്ന രീതികൾ ലഭ്യമല്ല. പൊതുവായി ലഭ്യമായ മലയാളം വാക്കുകളുടെ കയ്യെഴുത്ത വിവരശേഖരങ്ങളുടെ അഭാവം ഈ മേഖലയിൽ ആഴമേറിയ പഠിച്ചെടുക്കൽ രീതികൾ ഉപയോഗിക്കുന്നതിനു തടസ്സം ആവുന്നു. ഈ പഠനം സമഗ്രമായ ഒരു ഗവേഷണ ശ്രമത്തിലൂടെ പുതിയൊരു മലയാളം കയ്യെഴുത്ത വിവരശേഖരം ഉണ്ടാക്കിയും ആഴമേറിയ പഠിച്ചെടുക്കൽ അടിസ്ഥാനത്തിൽ കയ്യെഴുത്ത വാക്കുകളുടെ തിരിച്ചറിയൽ ഘടന വികസിപ്പിച്ചും അതുവഴി പ്രായോഗികമായ ഒരു രേഖകൾ കണ്ടെടുക്കുന്ന സമ്പ്രദായം ഉണ്ടാക്കിയും മേൽപ്പറഞ്ഞ പരിമിതിയെ അഭിസംബോധന ചെയ്യാൻ ഉദ്ദേശിച്ചുള്ളതാണ്.

ഈ പഠനത്തിൻറെ ഒന്നാം ഭാഗം മലയാളം കയ്യെഴുത്ത വാക്കുകളുടെ ഒരു വിവരശേഖരം ഉണ്ടാക്കുന്നതിൽ ശ്രദ്ധകേന്ദ്രീകരിച്ചിരിക്കുന്നു. അതിനാവശ്യമായ കയ്യെഴുത്ത ഇനങ്ങൾ ലഭ്യമാക്കിയത് പോലീസിന്റെ പ്രഥമ വിവര വിവരണങ്ങളിൽ നിന്നാണ്. വിവിധ പ്രായത്തിലുള്ള ആളുകൾ പകർത്തി എഴുതിയ മാതൃകകളും ഇതിനോട് കൂട്ടിച്ചേർത്തു. എല്ലാ കൈയെഴുത്ത രേഖകളുടെ ചിത്രങ്ങളും പ്രാഥമിക പ്രക്രിയകൾക്കു വിധേയമാക്കി പദങ്ങളുടെ ചിത്രങ്ങൾ ആക്കി മാറ്റി. പദസഞ്ചയത്തെ നിർവചിക്കാൻ കൂടുതലായി ആവർത്തിക്കപ്പെടുന്ന വാക്കുകളും പ്രഥമ വിവര റിപ്പോർട്ടുമായി ബന്ധപ്പെട്ട വാക്കുകളും ചുരുക്കപ്പട്ടിക ആക്കിയപ്പോൾ 321 പദവിഭാഗങ്ങൾ ലഭ്യമായി. ഓരോ പദവിഭാഗത്തിലും ഒരു നിശ്ചിത അളവ് ഇനങ്ങൾ ലഭ്യമാവാൻ 10 വയസ്സ് മുതൽ 65 വയസ്സുവരെയുള്ള വ്യക്തികളുടെ കയ്യെഴുത്തുകളും കൂടി ലഭ്യമാക്കി. ഈ സന്തുലിതമായ വിവരശേഖരം ആണ് തുടർന്ന് മലയാളം കയ്യെഴുത്ത വാക്കുകളുടെ തിരിച്ചറിയലിനുള്ള രീതികൾ വികസിപ്പിക്കാൻ ഉപയോഗിച്ചത്.

ഈ ഗവേഷണത്തിന്റെ രണ്ടാം ഘടകത്തിൽ ടി ആർ എ എം എച് ഡബ്ല്യൂ ആർ എന്ന് പേരിട്ടിരിക്കുന്ന നിർദ്ദേശിത കൈയെഴുത്ത് മലയാളം വാക്ക് തിരിച്ചറിയൽ ഘടന അവതരിപ്പിക്കുന്നു. ഈ ഘടനയിൽ താരതമ്യവിവേചനം പുറത്തെടുക്കുന്നതിനായി വ്യത്യസ്തവും ശ്രദ്ധാധിഷ്ഠിതവുമായ ഘടനകൾ സംയോജിപ്പിക്കുകയും ചെയ്യുന്നു. ഈ ഘടനകൾ ചിത്രങ്ങളിൽ നിന്നും സമൃദ്ധവും സന്ദർഭബോധമുള്ളതുമായ പ്രതിനിധാനം നൽകുന്നു. അതേസമയം വ്യത്യാസപ്പെടുത്തൽ ശേഷി കൂടുതൽ ശക്തമാക്കുന്നു. ഈ

സംയോജിത സമീപനം യഥാർത്ഥ ചിത്രങ്ങളിൽ സാധാരണയായി കാണപ്പെടുന്ന കൈയെഴുത്തിലെ വ്യത്യാസങ്ങളും കാഴ്ചയിൽ സമാനമായ വാക്ക് രൂപങ്ങളും കാര്യക്ഷമമായി കൈകാര്യം ചെയ്യാൻ മാതൃകയെ സഹായിക്കുന്നു.

ഈ പഠനത്തിന്റെ മൂന്നാമത്തെ സംഭാവന, വാക്ക് തിരിച്ചറിയൽ മാതൃകയുടെ ഫലം അടിസ്ഥാനമാക്കി വികസിപ്പിച്ച ഒരു പ്രായോഗിക രേഖകൾ കണ്ടെടുക്കുന്ന സമ്പ്രദായമാണ്. പ്രയോഗപരമായ ഈ ഘട്ടത്തിനായി ലഘുവായ തിരിച്ചറിയൽ ഘടന സ്വീകരിച്ചു. ശേഖരിച്ച കൈയെഴുത്ത് രേഖകളുടെ ഓരോ ചിത്രങ്ങളും മുന്നേ ഉപയോഗിച്ച പ്രാഥമിക പ്രക്രിയകൾക്കു വിധേയമാക്കി. 321 വാക്കുകൾ അടങ്ങിയ ചിത്രങ്ങൾ തിരിച്ചറിയാൻ ഇതിലൂടെ ഓരോ ഡോക്യുമെന്റിനും ഭാഗികമായ രേഖകളുടെ രൂപം ലഭിച്ചു. തിരിച്ചറിയപ്പെട്ട ഈ വാക്കുകൾ തന്നെ ഫലപ്രദമായ രേഖകൾ കണ്ടെടുക്കലിനായ മതിയായ വിവരങ്ങൾ നൽകുന്നു. തിരിച്ചറിയപ്പെട്ട വാക്കുകളിൽ നിന്ന് ഒരു പ്രതിനിധാനം സൃഷ്ടിക്കുകയും, രേഖകൾ തമ്മിലുള്ള അർത്ഥബന്ധം മനസ്സിലാക്കുകയും ചെയ്തു. ചോദ്യവുമായുള്ള സാമ്യം അടിസ്ഥാനമാക്കി ഓരോ രേഖയ്ക്കും സ്ഥാനം നൽകുകയും, ഏറ്റവും സാമ്യമുള്ള ഏറ്റവും മുകളിലുള്ള 5 രേഖകൾ കണ്ടെടുക്കൽ ഫലമായി നൽകുകയും ചെയ്യുന്നു.

ഈ ഗവേഷണം വിവരശേഖരനിർമ്മാണം മുതൽ തിരിച്ചറിയൽ മാതൃക വികസനം വരെയും കൂടാതെ രേഖകൾ കണ്ടെടുക്കുന്ന സമ്പ്രദായ വും ഉൾക്കൊള്ളുന്ന ഒരു സമ്പൂർണ്ണ രൂപഘടന അവതരിപ്പിക്കുന്നു. വികസിപ്പിച്ച രീതികൾ, കുറഞ്ഞ വിഭവങ്ങളുള്ള ഭാഷകളിലും ലിപികളിലും ഭാവിയിലെ ഗവേഷണങ്ങൾക്ക് ഒരു അടിസ്ഥാനമായി പ്രവർത്തിക്കുന്നു. കൂടാതെ കൈയെഴുത്ത് രേഖകൾ ഇപ്പോഴും പ്രധാന പങ്ക് വഹിക്കുന്ന സർക്കാർ നിയമ-ഭരണ മേഖലകളിൽ ഈ സംവിധാനങ്ങളെ വലിയ തോതിൽ പ്രയോഗിക്കാൻ വഴിയൊരുക്കുന്നു.

സൂചകപദങ്ങൾ: കൈയെഴുത്ത് മലയാളം പദം തിരിച്ചറിയൽ, ദർശന ട്രാൻസ്ഫോർമർ, രേഖകളുടെ ചിത്രങ്ങളുടെ വിശകലനം, വേഡ് സെഗ്മെന്റേഷൻ, ലോ-റിസോഴ്സ് ലിപികൾ, കൈയെഴുത്ത് പദ വിവരശേഖരം, രേഖകൾ വീണ്ടെടുക്കൽ, ശ്രദ്ധാ സംവിധാനം.

CONTENTS

Chapter No.	*Title*	*Page No.*
1	**INTRODUCTION**	**1-12**
	1.1 OVERVIEW OF HANDWRITTEN MALAYALAM WORD RECOGNITION	1
	1.2 BACKGROUND AND MOTIVATION	2
	1.3 CHARACTERISTICS OF MALAYALAM SCRIPT	5
	1.4 STATEMENT OF THE PROBLEM	6
	1.5 RESEARCH CHALLENGES AND SOLUTIONS	7
	1.6 OBJECTIVES OF THE STUDY	8
	1.7 THESIS OVERVIEW	9
	1.8 SUMMARY	11
2	**LITERATURE REVIEW**	**13-48**
	2.1 INTRODUCTION AND CHAPTER OVERVIEW	13
	2.2 SURVEY OF PUBLICLY AVAILABLE OFFLINE HANDWRITTEN WORD DATASETS	13
	2.3 SURVEY OF PREPROCESSING TECHNIQUES IN HANDWRITING RECOGNITION SYSTEMS	18
	2.4 SURVEY ON FEATURE EXTRACTION METHODS	23
	2.4.1 Handcrafted Feature Extraction Techniques	23
	2.4.2 Deep Learning based Feature extraction Techniques	29
	2.5 RECOGNITION TECHNIQUES OF HANDWRITTEN WORD IMAGES	32
	2.5.1 Classical or Heuristic Recognition Methods	32
	2.5.2 Machine Learning based Recognition Methods	33
	2.5.3 Deep Learning based Recognition Techniques	37
	2.6 DOCUMENT RETRIEVAL APPROACHES	44
	2.7 SUMMARY	47
3	**METHODOLOGICAL FRAMEWORK FOR DEVELOPING THE MALAYALAM HANDWRITTEN WORD DATASET (MHWD)**	**49-79**

	3.1	INTRODUCTION	49
	3.2	ACQUISITION OF HANDWRITTEN IMAGES FROM POLICE FIRST INFORMATION STATEMENTS (FIS)	51
	3.3	PREPROCESSING PIPELINE FOR ENHANCEMENT AND SKEW CORRECTION OF SCANNED HANDWRITTEN FIS DOCUMENTS	54
		3.3.1 Grayscale Conversion	54
		3.3.2 Gaussian Smoothing for Noise Suppression	55
		3.3.3 Local Contrast Enhancement Using Contrast-Limited Adaptive Histogram Equalisation (CLAHE)	56
		3.3.4 Morphological Opening to remove Isolated Noise Pixels	57
		3.3.5 Canny Edge Extraction to Support Skew Correction	57
		3.3.6 Probabilistic Hough Line Transform for Skew Correction	59
	3.4	WORD SEGMENTATION USING CONTOUR-BASED TECHNIQUES	64
		3.4.1 Binarization of Documents using Otsu's Method	64
		3.4.2 Morphological Dilation for word Connectivity Improvement	65
		3.4.3 Contour Detection for Word Localization	66
		3.4.4 Extraction and Saving of Segmented Word Regions	66
	3.5	SELECTION OF WORD CLASSES, WORD LABELING AND MHWD CONSTRUCTION	72
	3.6	DATASET CHARACTERISTICS AND STATISTICAL OVERVIEW	73
	3.7	SUMMARY	79
4	**TrA-MHWR: TRANSFORMER BASED WORD RECOGNITION MODEL WITH ATTENTION ENRICHED FEATURES**		**81-114**
	4.1	INTRODUCTION AND OVERVIEW OF TrA-MHWR MODEL ARCHITECTURE	81
	4.2	FEATURE EXTRACTION WITH DEIT	83
	4.3	FEATURE ENRICHMENT USING MULTI-HEAD SELF-ATTENTION AND RESIDUAL CONNECTION	86
		4.3.1 Multi-Head Self-Attention-Based Feature	86

Refinement
4.3.2 Residual Connection for Stability and Information Preservation 87
4.4 EXTRACTION OF THE CLASSIFICATION TOKEN AND WORD RECOGNITION VIA FULLY CONNECTED LAYERS 88
4.5 EXPERIMENTAL CONFIGURATION AND TRAINING SETUP 89
4.6 RESULTS AND DISCUSSION 91
4.6.1 Metrics for Assessing TrA-MHWR Model Performance 91
4.6.2 Quantitative Results and Discussion 93
4.6.3 Comparative Evaluation of Model Configurations 96
4.6.4 Benchmarking TrA-MHWR Model using Five-Fold Cross-Validation on the CMATERdb2.1.2 Dataset 108
4.6.5 Benchmarking With Existing Malayalam and Indic Handwritten Word Recognition Models 110
4.7 SUMMARY 113
5 DOCUMENT RETRIEVAL SYSTEM WITH LOW-RESOURCE-FRIENDLY MODEL SELECTION 115-135
5.1 INTRODUCTION AND SYSTEM OVERVIEW 115
5.2 PREPROCESSING AND WORD SEGMENTATION 118
5.3 EFFICIENTNET-BASED WORD RECOGNITION FOR EDGE DEPLOYMENT 118
5.3.1 EfficientNet-B0 based Feature Extraction 118
5.3.2 Recognition with FFNN 120
5.4 EXPLAINABILITY OF THE PROPOSED MODEL 122
5.5 METHODOLOGY FOR DOCUMENT RETRIEVAL SYSTEM 126
5.5.1 Document Corpus Preparation 126
5.5.2 Dictionary Construction and Bag-of-Words (BoW) Representation 127
5.5.3 TF-IDF Weighting Scheme 128

	5.5.4	Topic Modeling using LDA	129
	5.5.5	Document Topic Assignment	130
	5.5.6	Query Processing and Topic Identification	130
	5.5.7	Topic-based Document Filtering	131
	5.5.8	Calculating Similarity Based on TF–IDF and Cosine Similarity	132
	5.6	RESULTS AND DISCUSSION	132
	5.7	SUMMARY	135
6	**CONCLUSION**		**137-139**
	6.1	THESIS SUMMARY	137
	6.2	CONTRIBUTION OF THE THESIS	138
7	**RECOMMENDATIONS FOR FUTURE RESEARCH**		**141-148**
	7.1	INTRODUCTION	141
	7.2	DATASET EXPANSION AND DIVERSITY	142
	7.3	SEGMENTATION AND PREPROCESSING IMPROVEMENTS	143
	7.4	WORD RECOGNITION WITHOUT LEXICAL CONSTRAINTS	144
	7.5	POST-RECOGNITION ERROR CORRECTION	146
	7.6	CONTEXTUAL DOCUMENT RETRIEVAL ENHANCEMENTS	147
	7.7	CONCLUSION	148
	REFERENCES		**149-160**
	LIST OF PUBLICATIONS		
	CONFERENCE PROCEEDINGS		

LIST OF TABLES

Table No.	*Title*	*Page No.*
3.1	Segmentation Performance on Sample Documents	69
3.2	Word Segmentation Performance on Different Indian Scripts (Dataset: PHDIndic_11)	71
3.3	Statistics Summary of Image of Word Classes from the Handwritten Dataset (MHWD)	73
4.1	Final Test Performance Metrics of TrA-MHWR	95
4.2	Comparison with Different Model Configurations	107
4.3	Five-Fold Cross-Validation Results on Malayalam Word Dataset and CMATERdb2.1.2 Bangla Dataset using TrA-MHWR Model.	110
4.4	Comparative Performance of TrA-MHWR with Existing CNN–SVM and HWordNet Models	111
4.5	Comparison of state-of-the-art offline HWR studies and proposed work	112
5.1	Performance Metrics of the EfficientNet-B0 + FFNN Recognition Model	122
5.2	Extracted LDA Topics and Representative Keywords	133
5.3	Top five retrieved documents for the query with similarity score	134

LIST OF FIGURES

Figure No.	*Title*	*Page No.*
1.1	Offline Handwritten Word Recognition System	2
1.2	Malayalam Alphabet (Aksharamala)	5
1.3	Thesis Overview	9
3.1	System Architecture Diagram for MHWD Creation	51
3.2	Handwritten image samples of FIS document	52
3.3	Skew Correction Results	62
3.4	Overview of the preprocessing stages implemented for scanned handwritten documents	63
3.5	Block diagram illustrating the word segmentation pipeline using contour-based techniques with intermediate outputs at each step	68
3.6	Word segmentation results using traditional methods: Connected Components, Morphological Operations, and proposed method	70
3.7	Word segmentation results on handwritten document images from 9 Indian scripts and Roman Script	70
3.8	Word Samples Collected to make the Dataset Balanced	72
3.9	Swarm plot showing the distribution of average widths of handwritten Malayalam word images across different character counts	78
4.1	Architecture of the proposed TrA-MHWR model. It integrates the DeiT transformer for efficient feature extraction, followed by attention-based feature enhancement and classification using fully connected layers	82
4.2	Epoch-wise Test and Validation Accuracy of the TrA-MHWR model	94
4.3	Epoch-wise Training, Testing and Validation Loss of the TrA-MHWR model	95
4.4	Densnet with three dense blocks	98
4.5	MobileNetV2 Architecture	100
4.6	ResNet-152 Architecture	101
4.7	VGG19 architecture	103

5.1	Document Retrieval System Architecture	116
5.2	Recognition pipeline using EfficientNet-B0 for feature extraction and a fully connected layer for classifying handwritten Malayalam words	119
5.3	Classification report of the EfficientNet-B0 based Recognition Model	121
5.4	Grad-CAM explainability block diagram	124
5.5	Grad-CAM Results	125

LIST OF SYMBOLS AND ABBREVIATIONS

$B_i(x, y)$	:	Binarized image
ω_0, ω_1	:	Class probabilities
μ_0, μ_1	:	Class means
$\sigma_B^2(T)$	:	Between-class variance
θ^*	:	Skew angle
$P_\theta(i)$	:	Projection profile
x', y'	:	Transformed pixel coordinates after Arnold transformation
N_d	:	Image dimension (width or height) used in Arnold transform modulus
A_{nm}	:	Zernike moment of order n and repetition m
$V_{nm}(x, y)$	:	Zernike basis polynomial of order n and repetition m
a^*	:	Optimal LPP projection vector
Xd	:	Data matrix containing all feature vectors
Lp	:	Graph Laplacian matrix
Wp	:	Weight matrix encoding neighbourhood relationships between samples
Dp	:	Degree matrix
f_{global}	:	Global feature vector component
f_{local}	:	Local feature vector component
$A_k^{(l)}$	:	k^{th} feature map at layer l
$W_k^{(l)}$	:	k^{th} learnable convolutional kernel at layer l
$b_k^{(l)}$	:	Bias term
$\sigma_a(.)$	:	Non-linear activation function
X^{l-1}	:	Feature map from the previous layer
$D(i, j)$	:	Cumulative alignment cost
$d(.,.)$	:	Local distance measure
s_i	:	i^{th} input segment
t_j	:	j^{th} character template
$I(x, y)$	:	Gray scale Intensity of pixel coordinates
$R(x, y)$	:	Red channel intensity of pixel coordinates
$G(x, y)$	:	Green channel intensity of pixel coordinates
$B(x, y)$	:	Blue channel intensity of pixel coordinates
G_k	:	Gaussian kernel
σ	:	Smoothing Parameter
$I_s(x, y)$	:	Gaussian smoothed Image
h_k	:	Histogram count of grey level k
C	:	Clip limit

E	:	Clipped extra pixels
L_g	:	Number of gray levels
I_{in}	:	Input pixel intensity
I_{out}	:	Output pixel intensity
$\ominus$	:	Erosion
$\oplus$	:	Dilation
$\|G\|$	:	Intensity gradient magnitude
θ_i	:	Intensity gradient direction
$E(x, y)$	:	Edges
r	:	Perpendicular distance from line to the origin
θ	:	Orientation angle of the line
α	:	Skew estimation angle
t	:	Threshold
σ_0^2, σ_1^2	:	Variances
A	:	Binary Image
B	:	Structuring element
$(B)_z$	:	Translation of B by z
C_n	:	Contour
$N\ (x, y)$	:	Neighbourhood pixels
R	:	Bounding rectangle
I	:	Input image
H_{img}	:	Image height
W_{img}	:	Image width
Ch	:	Number of color channels
P	:	Patches
D	:	Embedding vector dimension
N	:	Number of image patches
X_p	:	Patch embeddings matrix
x_{cls}	:	Token Class
PE	:	Positional encoding matrix
L	:	Transformer encoder layer
Z	:	Embedding sequence
LN	:	Layer Normalisation
h	:	Attention head
Q	:	Query
K	:	Key
V	:	Value
W	:	Weight Matrix
C_{out}	:	Number of output classes
W_1, W_2	:	Weight Matrices

b_1, b_2	:	Bias
$\bar{y}$	:	Class probabilities
TP	:	True positive
TN	:	True negative
FP	:	False positive
FN	:	False negative
L_{CE}	:	Categorical cross-entropy loss function
h_t	:	Hidden state at time t
$h(x)$	:	Embedding of input image x
α_k^c	:	Weight importance
A_{ij}^k	:	Activation at spatial location (i, j) of the k^{th} feature map
$L_{Grad-CAM}^c$	:	Class activation map
Dc	:	Document Set
d_i	:	Ordered set of words
$\|v\|$	:	Vocabulary size
f_{ij}	:	Word frequency
$Topic_k$	:	Learned topic
$P(k \mid d_i)$	:	Posterior probability
$Topic(q)$	:	Dominant topics
$\mathrm{sim}(\mathrm{d}, \mathrm{q})$	:	Cosine similarity between vectors p and q
HR	:	Handwriting Recognition
OCR	:	Optical Character Recognition
HMWR	:	Handwritten Malayalam Word Recognition
FIS	:	First Information Statements
CNNs	:	Convolutional Neural Networks
RNNs	:	Recurrent Neural Networks
CTC	:	Connectionist Temporal Classification
ViT	:	Vision Transformer
HWR	:	Handwritten Word Recognition
LDA	:	Latent Dirichlet Allocation
MHWD	:	Malayalam Handwritten Word Dataset
Grad-CAM	:	Gradient-weighted Class Activation Mapping
RLSA	:	Run Length Smoothing Algorithm
FCM	:	Fuzzy C-Means
DoG	:	Difference of Gaussian
STN	:	Spatial Transformer Network
HOG	:	Histogram of Oriented Gradients
PHOG	:	Pyramid Histogram of Oriented Gradients
DCT	:	Discrete Cosine Transform
LPP	:	Locality Preserving Projections

PCA	:	Principal Component Analysis
SIFT	:	Scale-Invariant Feature Transform
SURF	:	Speeded-Up Robust Features
SCF	:	Statistical and Contour-Based Features
SOVQ	:	Self-Organizing Vector Quantization
HS	:	Harmony Search
SVM	:	Support Vector Machine
CDBN	:	Convolutional Deep Belief Network
CRBMs	:	Convolutional Restricted Boltzmann Machines
SGD	:	Stochastic Gradient Descent
HMM	:	Hidden Markov Model
MLP	:	Multi-Layer Perceptron
GA	:	Genetic Algorithm
DHBN	:	Dynamic Hierarchical Bayesian Network
CC	:	Connected Component
ELM	:	Extreme Learning Machine
FKNN	:	Fuzzy K-Nearest Neighbor
SMO	:	Sequential Minimal Optimization
LRM	:	Logistic Regression Model
NB	:	Naive Bayes
KNN	:	K-Nearest Neighbor
BLSTM	:	Bidirectional Long Short-Term Memory
GW	:	George Washington
CCA	:	Canonical Correlation Analysis
PHOC	:	Pyramidal Histogram of Characters
SGD	:	Stochastic Gradient Descent
AW	:	Anchor Words
IV	:	In-vocabulary
NAW	:	Non-Anchor Words
OOV	:	Out-of-Vocabulary
BiGRU	:	Bidirectional Gated Recurrent Unit
DBN	:	Deep Belief Network
EM	:	Expectation-Maximization
CRF	:	Conditional Random Fields
RBMs	:	Restricted Boltzmann Machines
MCDNN	:	Multi-Column Deep Neural Network
CER	:	Character Error Rate
LSTM	:	Long Short-Term Memory
HTR	:	Handwritten Text Recognition
IR	:	Information Retrieval
TF-IDF	:	Term Frequency-Inverse Document Frequency

DTW	:	Dynamic Time Warping
DCRB	:	District Crime Records Bureau
CLAHE	:	Contrast-Limited Adaptive Histogram Equalisation
HT	:	Hough Transform
PPHT	:	Progressive Probabilistic Hough Transform
NLP	:	Natural Language Processing
DeiT	:	Data-Efficient Image Transformer
AFEM	:	Attention-based Feature Enrichment Module
BERT	:	Bidirectional Encoder Representations from Transformers
ReLU	:	Rectified Linear Unit
AFFNN	:	Attention-based Feed-Forward Neural Network
FFNN	:	Feed-Forward Neural Network
MBConv	:	Mobile inverted Bottleneck Convolutional blocks
BoW	:	Bag-of-Words
GANs	:	Generative Adversarial Networks

1. INTRODUCTION

1.1 OVERVIEW OF HANDWRITTEN MALAYALAM WORD RECOGNITION

Handwriting Recognition (HR) is an important part of Optical Character Recognition (OCR) and a research field in pattern recognition and document image analysis. HR detects handwritten text from the scanned images and automatically transcribes it into a machine-readable format. HR can be done in two ways: online and offline. Online HR captures dynamic features that facilitate more accurate analysis and classification. Offline recognition operates on static scanned images of handwritten text that lack temporal features and thus faces more significant challenges due to variations in writing styles, document noise, skew, and degradation [1]. An HR control system is used to digitize and store records for better readability and storage. It plays a key role in applications such as word spotting and information retrieval from handwritten documents. So it makes archived content highly accessible. It speeds up administrative activities like form processing in the educational, government, and finance sectors. In addition, such handwriting recognition technology is used to improve digital accessibility for users who have difficulty using standard input devices and to enable them to interact more efficiently with computers.

Handwriting recognition is also important in automated assessment of handwritten answer scrpits, as handwriting recognition and semantic analysis models are able to assist in the interpretation of students' written answers, providing reliable scores, thereby improving assessment efficiency and consistency [2]. HR systems are composed of a structured pipeline with several intermediate steps that transform a raw handwritten input image into text. The process starts with image capture of handwritten document/word images that are acquired using scanners, mobile cameras, or other digital instruments. This raw input sample image is noisy, shows nonuniform lighting, distortion, and a few other artifacts. Thus, several preprocessing operations, including gray-level conversion, binarization, noise removal, normalization, and segmentation, are performed to improve the quality of

the text image and to extract word regions from it. After image preprocessing, the system executes feature extraction to produce discriminative representations of the handwritten word image. The features generated are then fed into the recognition process to assign the most likely word class. Finally, the system generates the recognized word output, which can be indexed, searched as a document, or fed to downstream NLP tasks. Every step in this process has a significant impact on recognition performance. The result should be the recognized word. The structure of the offline handwritten word recognition system is illustrated in Figure 1.1.

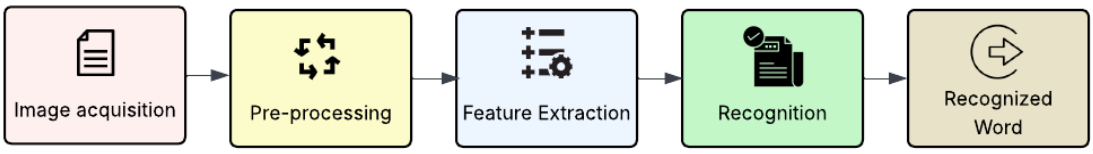

Figure 1.1 Offline Handwritten Word Recognition System

The scanned handwritten document image will be the input in offline Handwritten Malayalam Word Recognition (HMWR) system. After Image acquisition, the handwritten word recognition process goes through preprocessing, feature extraction and recognition stage. A classification model identifies the corresponding word. Recognition accuracy is influenced by the quality of input images, the effectiveness of feature extraction methods, and the robustness of the recognition model. Despite its practical significance in applications like archival digitization, forensic documentation, and administrative automation, HMWR remains underexplored due to a lack of standard datasets, script-specific segmentation tools, and efficient recognition models designed explicitly for Malayalam.

1.2 BACKGROUND AND MOTIVATION

The area of document image processing has been significantly growing, especially for handwritten documents, in the last few years. However, most of the advances have been predominantly focused on languages such as English, Chinese, and Arabic, while resource-scarce and script-intensive languages like Malayalam remain significantly underexplored [3]. Most of the existing work in Indic offline HR focuses on smaller linguistic units, such as characters and digits [4]. Although some research has been done on word-level recognition for scripts like Bangla and

Devanagari, the availability of annotated datasets and well-developed recognition systems is still inadequate for scripts like Malayalam, Gujarati, Oriya, and other Indian Scripts [5].

The analytical and holistic are the two main strategies employed for handwritten word recognition. The analytical approach breaks the words into individual characters before recognition, while the holistic approach views the word as a whole. In this research, we follow a holistic direction and handle the word image as a whole instead of breaking it into separate characters. For HMWR, the holistic approach offers a key advantage by avoiding the complex task of segmenting connected or compound characters, instead relying on the overall word shape for recognition [6]. The holistic approach makes it particularly effective in dealing with the Malayalam script's diverse and cursive nature, where character-level segmentation can be highly challenging. Processing Malayalam handwritten documents presents several challenges, including overlapping strokes, irregular spacing, the morphological tendency to form long word structures by adding multiple suffixes, and visually complex compound characters formed by merging various letters [7]. These issues illustrate the importance of research in this domain, especially for its practical application, such as the digitization of police records, legal documents, administrative papers, etc.

This work is motivated by the development task of processing the Malayalam handwriting recognition problem and the practical necessity for numerous fields such as legal systems, public administration, and cultural preservation. One of the core areas where automated Malayalam HR can have a transformative impact is in the processing of First Information Statements (FIS). FIS is a foundational component of criminal procedure. The FIS serves as the preliminary record of a criminal incident handwritten by police officers and forms the basis for the formal First Information Report (FIR). FIS documents are so frequently difficult to read due to the irregular and illegible handwriting. Handwriting quality variances often render the important information on the FIS document unreadable. This creates challenges for the Court because the FIS is the main handwritten record that forms the basis for later legal proceedings. This concern has been acknowledged in Circular No. 24/2019, dated 01/11/2020, issued by the Director General of Police in

Keralam, which mandates the immediate creation of a clear, typewritten version of each FIS to ensure accuracy and provide a reliable reference for the future. However, manually transcribing these documents is time-consuming and resource-intensive, highlighting the need for research into automatic HR systems to alleviate these challenges [8].

After the evolution of deep learning models, typical handwritten word recognition systems have adopted a hybrid architecture combining Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs) for feature extraction and sequence modeling. Besides, a Connectionist Temporal Classification (CTC) loss for alignment-free training and transcription [9]. Subsequently, an attention mechanism was introduced to improve input-output alignment, support end-to-end training with a hybrid loss function, and achieve competitive results in handwritten word recognition tasks [10]. Recently, transformer-based architectures, particularly Vision Transformer (ViT) like models, have been effectively utilized for handwritten text recognition, even in low-data scenarios [11], [12]. Their effectiveness for complex Indic scripts such as Malayalam remains underexplored. Therefore, an attention mechanism is introduced on top of the transformer-extracted features to strengthen feature discrimination and improve recognition performance for handwritten Malayalam words.

The research work is a stepping stone in advancing the theory and practice of handwritten word recognition on Malayalam, handling several important challenges associated with real-world document analysis systems. One of the great motivations in this task is that there is no perfect handwriting word recogniser available for Malayalam. The lack of available datasets is another indication of a gap in this research area. Accordingly, it is crucial to create a handwritten word dataset that can be used as both training/testing data for future model development and evaluation. This lack has hindered the research and production of strong recognition models for Malayalam. The distinct nature of the Malayalam script, which consists of a large number of characters in combination with complex ligatures and cursive style, poses problems for both character segmentation and recognition. Overcoming these challenges is crucial to make the digitization of legal documents, archival preservation of historical manuscripts, and automated processing of handwritten

forms feasible in practice. The growing importance of on-device computing in resource-constrained environments inspired the development of a lightweight yet effective recognition system.

1.3 CHARACTERISTICS OF MALAYALAM SCRIPT

The Malayalam script is derived from the Brahmi script and serves as a syllabic writing system primarily for the Malayalam language. It is also the native language of more than 40 million people [13]. The script evolved through various historical forms, including Vatteluttu and Grantha, and stabilized in its modern form around the 16th century. The Malayalam alphabet consists of 15 vowels and 36 consonants, along with numerous ligatures created from combinations of consonants, contributing to its visual and structural complexity. Figure 1.2 shows an overview of the Malayalam Alphabet (Aksharamala), including its vowels, vowel diacritics, consonants, consonant diacritics, chillus (half-consonants), and common consonant ligatures.

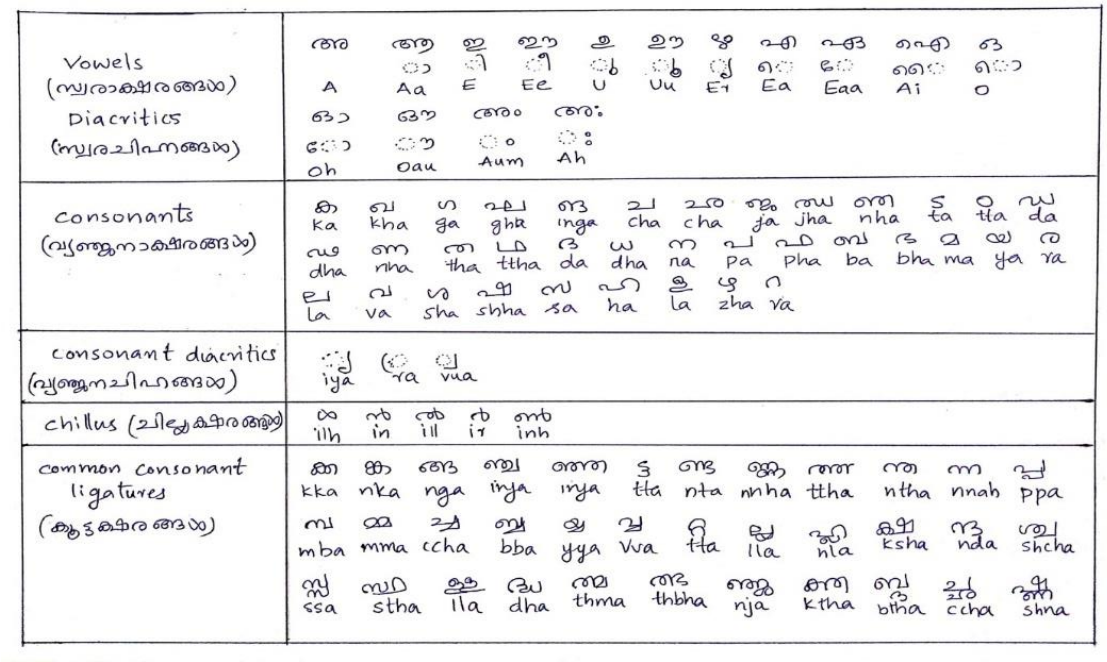

Figure 1.2 Malayalam Alphabet (Aksharamala)

It highlights the Malayalam script's complexity, particularly its use of diacritics and ligatures, essential for accurately representing the phonetic richness of the language.

The chart serves as a handy code reference for the purpose of linguistic analysis and text recognition.

1.4 STATEMENT OF THE PROBLEM

The digitization of handwritten Malayalam documents through scanning and archiving has become a common practice in recent years. However, the absence of robust automated recognition systems poses a significant limitation in converting these scanned images into searchable, machine-readable text. This shortcoming restricts efficient document retrieval and digital accessibility, particularly within legal and administrative domains.

A vast amount of handwritten Malayalam material is still being produced and used by schools, government offices, judicial institutions and police departments. Although such documents are often stored as digital images, their textual content remains inaccessible because current digitization processes do not extend to the recognition or indexing of the text.

Manual transcription of these manuscript records is still the most common way to create digital texts from these documents. Nevertheless, such a system is exhaustive and highly error-prone due to factors such as differing styles of handwriting, poor quality ink, and differences in spacing and alignment. These bottlenecks affect the retrieval of timely information, slow down the investigative process, and reduce possibilities for data-driven decision-making.

While notable advancements have been made in handwritten word recognition for Indian scripts such as Bengali and Hindi, research on the Malayalam script remains limited. The intricate morphology, compound character forms, and diverse handwriting patterns of Malayalam present additional challenges. Furthermore, the lack of a standardized, publicly available handwritten Malayalam word dataset restricts benchmarking and hinders the development of reliable recognition systems.

In summary, although digitization provides image-level preservation of Malayalam handwritten records, the absence of an automated handwriting recognition

framework prevents the extraction and retrieval of meaningful information. This gap highlights the need for designing an efficient and scalable handwritten word recognition system that can process real-world Malayalam documents and facilitate effective digital retrieval for legal and administrative applications.

1.5 RESEARCH CHALLENGES AND SOLUTIONS

Recognition of handwritten Malayalam words is a challenging task, especially while processing administrative and forensic documents like FIS. A significant challenge in preprocessing document images is the presence of noise, stains, faded ink, and uneven illumination. Scanning noise frequently occurs in digitized copies of handwritten records. Such noises significantly reduce the clarity of word boundaries, thereby affecting the accuracy of word segmentation.

The complex nature of the Malayalam script involving curves, ligatures, and stacked components is yet another challenge. These made the segmentation process of segmenting continuous handwriting into words extremely challenging, and features derived for recognition from these word images were not robust. In this study, a pipeline for noise reduction and contour-based segmentation was applied to avoid these problems. Such a preprocessing sequence improves the segmentation robustness and retains crucial stroke and spatial features of the extracted word images, which are necessary for recognition.

Beyond segmentation, the recognition stage posed additional challenges, especially due to variation in handwritten word shapes and inconsistent writing styles from writer to writer. To alleviate the above issue, we proposed a transformer-based attention and residual-enhanced model (TrA-MHWR) that provides further discriminative feature representations via both attention and residual connections. This feature helps the model learn detailed local information. Meanwhile, the global contextual relationships are utilized, which makes the recognition more robust under extreme visual conditions.

As an extension of the recognition framework, the study further developed a document retrieval system aimed at practical application in real-world environments.

Recognized words from handwritten documents are converted into machine-readable text and indexed for search. When a user provides a query sentence in Malayalam, the system employs Latent Dirichlet Allocation (LDA) to perform topic modeling and semantic matching, thereby retrieving documents that are contextually relevant to the query. This approach demonstrates how the proposed recognition framework can be effectively integrated into higher-level document analysis tasks such as content-based retrieval and knowledge discovery.

Overall, the study addresses both the foundational challenges of preprocessing and recognition in handwritten Malayalam text and the applied challenges of document retrieval, thereby contributing a comprehensive pipeline from raw handwritten document images to intelligent, searchable digital archives, suitable for administrative applications.

1.6 OBJECTIVES OF THE STUDY

The focus of this work is the development of an effective system for recognizing handwritten Malayalam words. Then extend the recognition model to apply it to document retrieval on real-world handwritten records. The task involves several difficulties, such as the unavailability of handwritten Malayalam word databases, complexity in script, and the requirement for an automated retrieval system of digitised handwritten documents. For this, the study identified the following specific objectives.

- Dataset Construction: To construct the handwritten Malayalam word dataset by extracting samples from real FIS. This dataset aims to ensure balanced representation across multiple word classes.
- Development of Recognition Model: The development of a model that is able to capture the structural and contextual information in Malayalam script with the help of a transformer-based architecture, which takes advantage of the attention mechanism and residual connections.
- Model Evaluation: To evaluate the comparative performance of the proposed model based on the constructed Malayalam Handwritten Word Dataset (MHWD) and comparison with other existing works and baselines for

recognition accuracy shows the superiority, robustness, and generalization capacity of the developed model.

- Framework for Document Retrieval: To integrate the recognition model outputs into a Latent Dirichlet Allocation (LDA)-based document retrieval system. This enables users to search and retrieve handwritten Malayalam documents efficiently based on sentence-level queries.

1.7 THESIS OVERVIEW

The thesis is organized in seven chapters, from basics to advanced ideas and practical techniques for offline HMWR and document retrieval. A general schematic of the thesis is provided in Figure. 1.3.

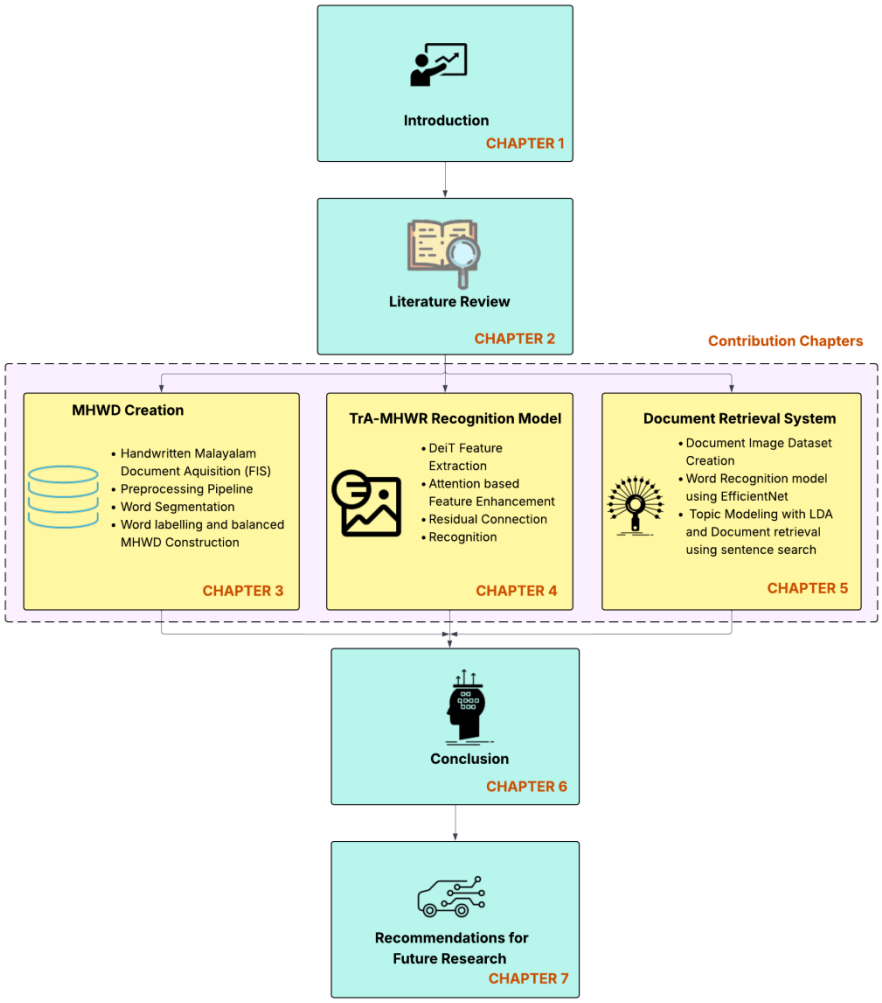

Figure 1.3 Thesis Overview

Chapters 1 and 2 address the opening parts of the thesis. It concentrates on the research context and background, with a thorough review of related work. Chapters 3, 4, and 5 contribute to the three innovation pillars of the research: dataset construction (Chapter 3), model development (Chapter 4), and Document Retrieval System implementation (Chapter 5). Finally, Chapter 6 presents the study's conclusions, while Chapter 7 outlines recommendations and directions for future research.

Chapter 2 includes a review of existing studies in handwritten word recognition. It begins with an overview of notable handwritten word datasets and outlines standard preprocessing techniques performed on handwritten data. The chapter then provides a review of feature extraction methods, categorized into handcrafted, machine learning-based, and deep learning-based approaches. Offline Handwritten word recognition strategies, including machine learning and deep learning models, are reviewed and discussed. Moreover, an overview of document retrieval systems designed for Indic scripts and other languages is included. This chapter explains the work done in earlier studies and identifies the areas that remain unaddressed. It presents the research gaps that created the need for the present work.

Chapter 3 provides the first significant contribution of the thesis. It focuses on the different phases of developing the offline handwritten MHWD. It begins by describing the collection of handwritten document images, specifically sourced from police FIS. The chapter details the preprocessing techniques employed to enhance image quality, followed by applying contour-based segmentation methods for extracting individual word images from the document image. The study also describes the labelling process and construction of the whole dataset, constituting an organized dataset. Next, the content and diversity of the dataset are presented. The chapter ends with a holistic summary of its contributions to the study.

Chapter 4 details TrA-MHWR Recognition model which is the second contribution. TrA-MHWR is a transformer-based model designed for handwritten word recognition. It introduces the motivation behind developing this model and highlights its relevance to the challenges in recognizing handwritten Malayalam

words. The chapter uses the Data-efficient Image Transformer (DeiT) framework for feature extraction and model enhancements using attention mechanisms and residual connections. The recognition pipeline is then explained, outlining how the optimized model processes input features. Experimental settings are described, along with a thorough analysis of the results. Comparative evaluations with alternative recognition models and feature extraction techniques are also explained to demonstrate the effectiveness of TrA-MHWR. The chapter concludes with a summary of the findings and their significance.

Chapter 5 extends the handwritten word recognition system into a document retrieval framework. After preprocessing and segmentation, each handwritten page was converted into digital text using an EfficientNet-B0 and FFNN-based recognition model, resulting in a searchable document dataset.EfficientNet-B0 was selected for this stage due to its suitability for edge deployment while maintaining high recognition accuracy. Gradient-weighted Class Activation Mapping (Grad-CAM) was employed to interpret the model's decisions and ensure explainability visually. To enable retrieval, LDA was used to generate topic-based representations of all documents. A Malayalam sentence query is processed through the same pipeline, and similarity scores are computed to return the top five most relevant documents. This chapter describes the retrieval system, demonstrating its potential for practical use in digital access and search applications.

Chapter 6 concludes the research by presenting the summary of the major contributions and impact of the proposed handwritten Malayalam word recognition and document retrieval framework.

Chapter 7 provides future research directions, outlining possible improvements, extensions, and opportunities for further exploration based on insights gained from this work.

1.8 SUMMARY

This chapter provides the motivation, background, and research significance in the development of an automatic handwritten Malayalam word recognition system and

a scanned handwritten document retrieval system. It began by outlining the continued reliance on handwritten Malayalam documents within administrative and legal domains, highlighting the challenges associated with digitising such documents due to handwriting variability, limited annotated resources, and the absence of standardised recognition systems for the Malayalam script. The chapter then emphasised the research gap stemming from the lack of publicly available handwritten word datasets and the limited exploration of transformer-based architectures for recognition tasks in complex Indic scripts. Based on these limitations, the need for a robust pipeline encompassing dataset creation, reliable recognition, and search-based retrieval was identified. The key objectives were formulated accordingly, focusing on dataset development, attention-enhanced recognition modelling, model evaluation, and retrieval-based application. The chapter provided an overview of the complete thesis structure. It offered a clear roadmap that connected each subsequent chapter to the overall research problem.

2. LITERATURE REVIEW

2.1 INTRODUCTION AND CHAPTER OVERVIEW

Handwritten word recognition is an important topic in pattern recognition and document image analysis. It has been increasingly applied to digitization, archives, and information retrieval. This chapter reviews related literature for offline handwritten word recognition systems. It starts by introducing offline handwritten word public datasets in several scripts and reveals how they are scarce for the Malayalam language. The next section will concentrate on preprocessing methods applied to handwritten images, which are critical to the effectiveness of subsequent stages of handwriting recognition. It then describes various feature extraction strategies, which are classified into handcrafted methods, machine learning, and deep learning based techniques. After this, recognition approaches are surveyed, including traditional machine learning classifiers and more recent deep learning networks used in some works.

Apart from word recognition, in this chapter, we present handwritten retrieval systems that operate under a search-by-query logic. It is significant for the practical work of digital storage, administrative record management, etc. The chapter concludes with an overview of the most important results and open research challenges, which guided this research.

2.2 SURVEY OF PUBLICLY AVAILABLE OFFLINE HANDWRITTEN WORD DATASETS

Access to handwritten datasets is fundamental for research in numerous applications, such as handwritten recognition, writer identification, keyword spotting, and document analysis. These datasets are used for machine-learning and deep learning research and have been cited in peer-reviewed academic journals. A good dataset should model the broad range of variations introduced by different handwriting styles and scripts, and take into account the effects of document conditions.

Over the years, several offline handwritten datasets have been developed for different scripts. Several datasets exist for the Latin script, as outlined below. An early English handwriting dataset to recognize addresses, comprising 5,000 handwritten samples each of city names, state names, ZIP Codes, and alphanumeric characters. It is created using real-world mail data scanned at high resolution to ensure natural handwriting variability [13]. The IAM Handwriting Dataset is one of the most widely used and publicly available resources for handwritten English text recognition. The dataset contains 1,066 forms written by approximately 400 individuals. It contains 82,227 handwritten word instances derived from a vocabulary of 10,841 unique words [14].

The RIMES dataset was developed to support the evaluation of handwriting recognition and indexing systems for French handwriting applications. It comprises 12,723 pages from 5,605 letters contributed by over 1,300 participants. While developing dataset, volunteers were asked to compose letters based on realistic scenarios, such as address changes, complaints, and payment issues. The dataset has been used as a benchmark in several international competitions such as ICFHR 2008, ICDAR 2009 and ICDAR 2011. [15], [16].

The CENSUS-HWR forms one of the largest english handwritten word recognition database. It includes more than 1.8 million grayscale images. The corpus is taken from the 1930 and 1940 U.S. Census forms, contains texts composed by approximately 70,000 enumerators each year, and consists of a vocabulary with 10,711 words. This extensive data set is commonly used as a benchmark to test deep learning models for handwriting recognition [17]. The CVL database includes 101,069 written words in German and English. It is acquired from scanning the 311 author RGB scans. Word annotations and writer IDs based on XML are available for each sample. Its bilingual nature also makes this collection suitable for writer retrieval and multilingual word spotting research [18]. The IIIT-HW-English-Word dataset is collected from 20,800 A4 pages of English writing done by 1,215 individual, extracted words are about 757830 words. It was acquired by means of different mobile devices and in a variety of conditions, which replicates the real

process of digitization. Dataset contains stopwords as well as numerals, which can be used for tasks like form processing. It is also highly lexically diverse with 174,701 unique words and rare words are very frequent compared to the normal words [19].

Below is a list of significant offline handwritten Arabic word datasets, including contemporary and historical documents. The IFN/ENIT Corpus is a widely used benchmark Arabic handwritten word recognition database. It comprises 26,400 words used for the encoding of Tunisian localities as well as more than 210 k characters. This is taken from 411 authors via structured forms and scanned at 300 pixels per inch. They also contain word boundary, baseline and character structure annotations and the writing style. IFN/ENIT dataset was employed in Arabic HWR research and international competitions (e.g ICDAR 2005, ICFHR 2006) [20]. The KHATT dataset consists of 1,000 full page handwritten Arabic documents including more than 1,000 writers from over 50 countries. The images have been scanned at resolutions from 200 to 600 dpi, and are manually annotated for page, paragraph, and line boundaries. The database is employed for multi-resolution recognition and writer identification studies [21].

The AlexU-Word database includes 25,114 (written) Arabic word samples written by 907 writers from a controlled set of 109 words that represent all letter shapes. Annotations for word labels and stroke order are provided. It therefore provides a useful framework for developing and comparing segmented letter recognition systems [22]. The Muharaf dataset features more than 1,600 high-quality scanned pages of ancient Arabic manuscripts dating back to the 19th and 20th centuries from personal and legal documents. This query contains some annotations such as: coordinates on the polygonal line and classification of page elements (margins, stamp). The developed dataset comprises more than 12k lines text and is useful in cursive handwriting recognition, as well as paleography studies [23].

Moreover, the Chinese script delivers benchmark datasets in terms of handwriting recognition. The CASIA-HWDB 2.0, 2.1 and 2.2 datasets. It includes offline handwritten text which is written by 1,020 writers in Chinese characters. The

dataset includes more than 5,000 scanned pages with some ~1.35M character samples from different sources like news articles, poems and daily expressions. The data is annotated at character, word and line levels accompanied with segmentation ground truth. These resources have been used for a range of research tasks variously including character recognition, text line segmentation, document retrieval and writer identification [24]. The HIT-MW dataset is a corpus of offline handwritten text lines in Simplified Chinese script, which is composed of different writers. It consists of 853 scanned pages and a total of 186,444 characters are manually tagged at both the word and character levels. The data were collected by acquisition of handwriting during controlled writings of text scanned at 300 dpi. This dataset supports research in handwritten text line recognition and word-level segmentation [25].

Another script provides offline handwritten datasets in Cyrillic, focusing on Russian and the Cyrillic variant of Kazakh. The HKR dataset comprises offline handwritten samples in Cyrillic script, primarily in Russian 95% and Kazakh. It includes over 1,400 handwritten forms containing around 63,000 sentences and more than 715,000 symbols from approximately 200 writers. The dataset supports sentence, word, and character-level segmentation and covers standard Cyrillic characters and additional Kazakh-specific letters. In this LaTeX-generated forms are used for data collection and the dataset is suitable for word and line-level handwriting recognition research [26].

The CArDIS dataset focuses on historical Swedish handwriting and includes 116,000 single-character images and 30,000 handwritten Swedish names, extracted from over 64,000 historical documents. It provides character and word-level data captured using various inks and writing instruments, making it valuable for historical document analysis and recognition research [27].

Some benchmark word-level datasets are available for Indic scripts, supporting research in handwritten text recognition. The IIIT-INDIC-HW-WORDS dataset contains 872,000 handwritten word images across eight Indic scripts, including Bengali, Gurumukhi, Kannada, Odia, Gujarati, Malayalam, Urdu and Tamil scripts.

Collected from 135 writers using structured forms with 500 common words per script, it includes word-level bounding boxes and Unicode transcriptions. This dataset supports multilingual handwritten text recognition and cross-script research [28]. The IIIT-Indic-HW-UC dataset consists of 2.6 million handwritten word images over 13 Indic scripts which have been scanned from 1,220 writers. The data collected with mobile phones under diverse conditions is unconstrained, real-world handwriting with word-level annotations [29].

The BN-HTRd dataset contains 108,147 handwritten Bangla word samples, including 23,115 unique words, collected from 150 writers. Sourced from scanned news articles and paragraphs, it offers document, line, and word-level segmentation, making it valuable for Bangla handwritten document digitisation and recognition research [30]. The GHWD dataset includes 62,000 handwritten Gurmukhi word images collected from 40 writers. Derived from structured land record forms, it features administrative names such as tehsil and sub-tehsil labels. The dataset supports research in geo-name recognition using the Gurmukhi script for Punjabi language [31]. The CMATERdb 2.1.2 dataset comprises 18,000 grayscale images of handwritten Bangla city names, covering 120 unique classes with 150 samples each, at 300 dpi scanned images. Designed for holistic word recognition, the dataset avoids character-level segmentation and is freely available for research in Bangla handwriting recognition [32].

While different offline handwritten word datasets are available for major scripts like Latin, Chinese, and other Indic languages, they primarily focus on general-purpose content and high-resource languages. The limited number of existing Malayalam datasets often suffers from small scale, insufficient lexical diversity, or restricted public accessibility. Moreover, domain-specific datasets, particularly those based on legal or administrative documents, are scarce across different scripts. This gap limits the development and evaluation of recognition systems and other research works in Malayalam script. Hence, it is very essential to create a publicly accessible handwritten Malayalam word dataset that is both diverse and carefully annotated.

Also its remaining issues, such as complex problems of the script have to be supported and accomplished for practical use.

2.3 SURVEY OF PREPROCESSING TECHNIQUES IN HANDWRITING RECOGNITION SYSTEMS

Preprocessing is a necessary phase in handwritten text recognition. It improves the quality of the images, standardizes the data used as input to obtain a consistent feature extraction, and increases recognition accuracy as a whole. The basic preprocessing techniques are: image enhancement, binarisation, removal of noise and the skew detection, skew correction and segmentation of the written text into important components for analysis at a later stage [33]. Kim and Govindaraju introduced a preprocessing method to represent word contours in 8-directional Freeman chain codes for an efficient and compact shape encoding, where each contour pixel is encoded by a direction value $d \in \{0,1,2,\dots,7\}$ representing the eight neighbouring directions. The stroke thickness was averaged by horizontal contour tracing to allow adaptive noise reduction and improved feature extraction. Regarding segmentation, they extracted up to four candidate points for each character designed to segment touching characters minimizing the risk of over-segmentation. However, the method was highly dependent on accurate contour extraction and was sensitive to handwriting variations and broken character strokes [34].

Dasgupta et al. used preprocessing to correct the slant and gradian of handwritten words. Then the images were reshaped into square matrices to meet the requirement of Arnold transform, which performs periodic pixel shuffling, defined as shown in Eq. (2.1) for structural pattern representation.

$$(x', y') = (x + y, x + 2y)\ mod\ N \tag{2.1}$$

In the end, morphological opening and closing operations were conducted to reduce the image noises. The method was sensitive to normalization accuracy, and morphological processing could affect fine handwritten stroke details in degraded images [35]. Jayech et al. performed some preprocessing steps such as image thinning, removing horizontal and vertical spacing, diacritical marks deletion. They

utilized non uniform strategy with vertical histogram projection for the splitting of every word in its character by analysing pixel density variations along the vertical direction as shown in Eq. (2.2).

$$H(x) = \sum_{y} B_i(x, y) \tag{2.2}$$

Where $B_i(x, y) \in \{0,1\}$ is the binarized image. The approach was sensitive to irregular character spacing and overlapping handwritten strokes, which could reduce segmentation accuracy. [36].

Roy et al. utilized global histogram Otsu's method for binarization of handwritten document images, where an optimal threshold T^* value was selected by maximizing inter-class variance between foreground and background pixels as shown in Eq. (2.3).

$$\sigma_B^2(T) = \omega_0(T)\omega_1(T)[\mu_0(T) - \mu_1(T)]^2 \tag{2.3}$$

Where ω_0, ω_1 are class probabilities and μ_0, μ_1 are class means. Line segmentation was performed in a morphological manner, based on the foreground seed components and boundary information for separating text line by text line. Word space detection was based on Run Length Smoothing Algorithm (RLSA). RLSA removes horizontal pixels runs in the binarized text image that would join characters within a word and separate individual words as words, sorting them separately. The water reservoir technique was used for skewness correction, and slant correction was achieved by vertical projection histograms in combination with the Wigner–Ville distribution. The performance of the method was dependent on accurate binarization and could be affected by complex document degradations and irregular handwritten spacing [37]. Bhowmik et al. used disk filters for smoothing the handwritten word images and then applied binarisation based on Otsu global histogram. Although the smoothing operation reduced image noise, excessive filtering could suppress fine handwritten stroke information and affect recognition accuracy. [32].

Singh et al. utilised preprocessing methodologies for the improvement of handwritten word images that includes slope correction, slant correction, space compaction and size normalisation. Slope regulation was realized by calculating the

inclination angle of baseline and rotating word image to counterpose the direction of it. Shear correction was applied with regard to slant. In order to compact the space, no-foreground-column was removed. To remove salt-and-pepper noise, morphological operations (open and close) were implemented. For the sake of retaining and highlighting the underlying structural elements in handwriting, employed skeletonisation. The effectiveness of the method depended on accurate baseline estimation, and excessive thinning could distort fine handwritten stroke details [38]. Kadhm et al. proposed a preprocessing method that convert the gray-level text images were converted into binarized using Fuzzy C-Means (FCM) thresholding method, which groups pixels based on membership values rather than fixed threshold boundaries. After binarisation, image thinning was used to maximum stroke width into one pixel and normalisation was conducted to maintain uniform dimensions of data for computation. The approach was sensitive to noise and intensity variations, which could influence clustering accuracy during threshold selection [39].

Hegadi et al. used the method of Otsu's thresholding to binarise handwriting images. Noise reduction was then carried out using median, linear and adaptive filters. Moreover, we conducted skew correction for correctly arranging text, and corner points detection was completed as a means of structural analysis [40]. Al-Nuzaili et al. utilized sophisticated and simple universe of discourse models for preprocessing to better capture the pixel-level context. The work applied this method to assist the feature extraction for pixel positions, aiming at better representation of handwritten word images. The approach required careful parameter selection, as variations in handwriting density and image quality could affect contextual modelling accuracy [41]. Malakar et al. less noise via a set of preprocessing operations including image binarisation, morphological operations for structure refinement and Gaussian smoothing to suppress noise [42].

Stauffer et al. applied Difference of Gaussian (DoG) filtering and binarisation to improve the input image at first during preprocessing step, where DoG filtering

emphasizes edge information by subtracting two Gaussian-smoothed versions of the image at different scales as shown in Eq. (2.4).

$$DoG(x, y) = G(x, y, \sigma_1) - G(x, y, \sigma_2) \tag{2.4}$$

Further processing was carried out, as word segmentation, connected component labelling and skeletonisation were used to extract a handwritten word image for analysis. The method was sensitive to parameter selection in DoG filtering and could produce fragmented components in degraded handwritten images [43]. Zaghdoudi et al. performed a set of preprocessing operations on the binarised word images of from the IFN/ENIT database. It has pre-processing stages including slope and slant correction to correct the writing distortions, skeletonisation that reduces stroke width of the characters while preserving structural details, and normalisation which standardize image size for intension analysis [44]. Sawant et al. hand-cropped character images, and normalized them before applying binarisation with global thresholding. Other pretreatment procedures they used were edge detection, morphological dilation and fill of region in order to emphasize the structural description of handwritten characters. The method relied heavily on manual preprocessing and could exhibit reduced scalability for large handwritten datasets [45].

Patel et al. preprocessed the input image based on estimated baseline, skew and slant correction and horizontal/vertical scaling. Subsequently, the skeletonization step was applied to limit the number of character strokes for further analysis. The effectiveness of the method depended on accurate skew and baseline estimation, while excessive thinning could affect stroke continuity [46]. Sushma et al. used a series of preprocessing steps that consists of image binarization, line and word segmentation to assist in the recognition handwritten Kannada words. Variations in handwriting alignment and irregular word spacing could influence segmentation accuracy and subsequent recognition performance [47]. Gaur et al. used vectorization with the application of well-controlled distortions to the input images for preprocessing prior knowledge training. Although the distortion process increased data variability, excessive transformations could alter important structural

properties of handwritten characters [48]. Bhowmik et al. noise reduction by a disk filter and binarization with the use of such an elementary (simple) thresholding method. Erosion and dilation were applied to remove isolated foreground pixels and smooth the contours for better image quality. The approach was sensitive to threshold selection, and repeated morphological operations could distort fine handwritten stroke details [49]. Thomas et al. applied preprocessing techniques to correct skew and slant using respectively projection profile method and moment-based approach. In projection profile method the skew angle is calculated as shown in Eq. (2.5).

$$\theta^* = argmax_\theta \sum_i [P_\theta(i)]^2 \tag{2.5}$$

Then conducted contour refinement, aiming to enhance the structural coherence of handwritten word images. The method was sensitive to inaccuracies in skew estimation, particularly for highly cursive handwriting [50].

Balci et al. employed different preprocessing and data augmentation techniques on the dataset, including image padding, rotation, and zero-centering, which standardize the input data [51]. Dutta et al. introduced a preprocessing pipeline involving slant and slope correction followed by normalization. Besides, data augmentation is also employed using affine transformations, elastic distortions, and multi-scale transformation strategies. Their model also incorporated a Spatial Transformer Network (STN), which learns geometric transformations automatically to correct input distortions during training. The effectiveness of the method depended on the quality of learned spatial transformations and increased computational complexity [52]. Krishnan et al. used a method for generating synthetic handwritten word images using publicly available font classes and developed IIIT-HWS dataset, which contains 9 million synthetic samples to support large-scale deep learning. Synthetic generation improved data availability, although artificially generated handwriting may not fully capture natural writing variability [53]. Wigington et al. utilized augmentation techniques including grid-based distortion and test-time geometric augmentation to enhance model robustness. Grid distortion modifies local spatial regions to simulate realistic handwriting deformations, but excessive augmentation

could introduce unrealistic patterns [54]. Wicht et al. employed global thresholding, normalization, and image resizing during preprocessing and used a horizontal sliding window approach for extracting image patches. The sliding window approach captures sequential local features from handwritten words, although fixed window sizes may limit adaptability to varying character width [55]. Wu et al. employed data augmentation strategies, including rotation, shearing, and zooming, to enhance the variability of training samples and improve generalization performance. While augmentation enhanced robustness, large geometric transformations could distort important structural characteristics of handwritten words [56].

The reviewed studies show that pre-processing is an essential stage in handwritten word recognition, as it directly influences the effectiveness of subsequent segmentation and feature extraction processes. Existing methods commonly employed techniques such as thresholding, morphological processing, normalization, filtering, skew and slant correction, and skeletonization to enhance handwritten document quality. Nevertheless, many approaches remain sensitive to handwriting variability, degraded document conditions, broken strokes, irregular spacing, and complex character structures frequently observed in Indic scripts. Considering these limitations, the proposed work adopts a preprocessing strategy tailored for scanned Malayalam handwritten FIS documents to improve structural continuity and enhance the quality of handwritten word segmentation.

2.4 SURVEY ON FEATURE EXTRACTION METHODS

The process of feature extraction is among the essential part in handwritten word recognition systems. It decides how well the visual appearance and script-dependent structures of word images are captured. In this section, we study the two major classes of existing feature extraction methods in prevalent works: Handcrafted methods and deep learning-based approaches.

2.4.1 Handcrafted Feature Extraction Techniques

The conventional handwritten word recognition methods were mostly based on handcrafted features, which would be manually designed to describe visual cues in

the input image (e.g., shape, edge direction and pixel distribution). Commonly employed methods were Histogram of Oriented Gradients (HOG), zoning techniques, chain code representations, projection profiles and contour based descriptors [57]. However these techniques rely strongly on the domain knowledge and usually fail to cope with variations in handwriting style and noise. This makes it not sufficient powerful for scripts like Malayalam. Kim and Govindaraju also extracted linear operators in chain-coded segments to obtain global features of 3×3 subregion, which model directional gradients and topology patterns. Subsequently, structural units including loops and junctions were sampled from segmented areas of the input image to encode fine-grained character properties. The approach depended heavily on accurate segmentation and contour representation, which could be affected by highly cursive handwriting [34].

Dasgupta et al. proposed methods of feature extraction, aimed at representing the directional stroke patterns in handwritten word images. The approach employs Arnold transform as an orientation pooling function on scaled images to produce multiple transformed ones (Arnold planes) for better resolution of orientations as shown in Eq. (2.6).

$$(x', y') = (x + y, x + 2y) mod\ N_d \tag{2.6}$$

Strokes are connected and the stroke orientation distribution is determined, using Hough transform on each plane. In sub-regions Arnold transform is also utilized to encode local and global features, which makes multi-level feature extraction possible. The last feature vector is obtained by concatenating the directions of all levels and gives a detailed (yet expensive, in terms of computation) representation. Although the multi-level representation improved directional feature analysis, the approach involved high computational complexity [35]. For feature discriminatory measure, Jayech et al. used Zernike and Hu moment invariants to preserve the structure based features of handwritten characters as shown in Eq. (2.7).

$$A_{nm} = \frac{n+1}{\pi} \int_{-1}^{1} \int_{-\sqrt{1-x^2}}^{\sqrt{1-x^2}} f(x,y)[V_{nm}(x,y)]^* dydx \tag{2.7}$$

Then, for the more compact representation of extracted features, vector quantisation using the k-means clustering algorithm was employed [36]

Roy et al. proposed a zone-wise word segmentation where the image for each word is segmented based on three zones (upper, middle, and lower). Pyramid Histogram of Oriented Gradients (PHOG) was used to extract feature from middle zone and captured multi-resolution gradient orientation features to effectively express the structural information of handwritten text [37]. Bhowmik et al. proposed a shape-based feature descriptor with various pixel density histograms such as tetragonal, elliptical and vertical histograms combined to describe the structural properties of handwritten word images. Two other characteristics referred to the number of peaks and valleys (extracted from pixel density histogram along vertical dimension) to identify common shape variations in word image structure. The method was sensitive to writing style variations and irregular word structures [32]. Kadhm et al. proposed a recognition model for Arabic handwritten words based on the AHDB dataset. Structural and statistical properties were included in the feature extraction step such as connected component and also zoning-based descriptor. Global transform features (Discrete Cosine Transform (DCT) and HOG) were also extracted to represent discriminative features. Then the extracted feature set was further normalised. The performance of the method depended on effective feature normalization and segmentation quality [39].

Hegadi et al. used the Locality Preserving Projections (LPP) method to extract discriminative features and reduce dimensionality, where LPP finds a projection as shown in Eq. (2.8).

$$a^* = argmin_a \frac{a^T XdLpXd^T a}{a^T XdDpXd^T a} \tag{2.8}$$

preserves the local neighbourhood structure in a low-dimensional subspace, where $Lp = Dp - Wp$ is the graph Laplacian, Wp is the weight matrix and Dp is the degree matrix. The method was sensitive to neighbourhood parameter selection and could be affected by noisy handwritten samples [40]. Al-Nuzaili et al. introduced a pixel distribution feature extraction technique, and is underlined by the generation

of four kinds of features with the angular span, distance span approach and horizontal and vertical spac- ing method. These characteristics were combined in single, double, triple or quadratic fashion for assessing their effect on recognition rate. Three configurations which produced the highest mean recognition rates were chosen for further experimental validation. The approach required careful feature combination selection to avoid redundancy and increased computational complexity [41].

Malakar et al. utilized various elliptical and gradient methods for global and local features extraction in the feature extraction stage. This assumption is made since local features might include redundant or irrelevant data; thus, feature selection was performed to purify the local descriptors. The new local features were pooled together with the elliptical and gradient based local features to generate a compact local representation. Finally, the selected Global, Optimized Local Features were concatenated using feature selection to form the final features set for recognition purpose as shown in Eq. (2.9).

$$f = [f_{global} || f_{local}] \tag{2.9}$$

Although feature optimization reduced redundancy, the selection process increased computational overhead [42]. Stauffer et al. introduced six distinct graph extraction techniques for handwritten word images. It uses the keypoint detection, segmentation grids, projection profiles and image splitting. The segmentation grid-based method was divided into three methods, namely Grid Neighbourhood Analysis, Grid-based Minimum Spanning Tree, and Grid Delaunay Triangulation. Each contributed to the representation of different structural relationships in the grid layout. These representations effectively captured structural dependencies, although the methods were computationally intensive for complex handwritten patterns [43].

Zaghdoudi et al. used DCT and HOG as a feature extractor. In order to decrease the high dimensionality of HOG feature descriptors, Principal Component Analysis (PCA) was used for a more compact and efficient representation of features. Although PCA improved feature compactness, dimensionality reduction could lead to loss of discriminative information [44]. Sawant et al. used edge detection and

endpoint analysis for structural feature extraction of handwritten characters for recognition. Endpoint analysis identifies stroke termination points to represent character structure, though the method was sensitive to broken strokes and image noise [45]. Patel et al. pulled zone-based structure features from the image. It finds loops, junction points and end points in an image to constrain the geometric/topological properties of the handwritten script [46]. Sushma et al. extracted features by Scale-Invariant Feature Transform (SIFT) and Speeded-Up Robust Features (SURF) along with local descriptors to record invariant properties extracted from handwritten word images. SIFT and SURF detects stable local keypoints based on gradient distributions, providing robustness against scale and rotation variations. However, the extraction process was computationally intensive for large datasets [47]. Bhowmik et al. used concentric rectangle areas and convex hull to extract features, focusing on spatial and shape information of handwriting words images. Convex hull analysis represented the outer boundary structure of handwriting, although irregular word shapes could affect feature consistency [49]. Thomas et al. extracted features using pixel density distribution and concavity to capture the structural properties of script. Concavity features capture inward structural variations of handwritten shapes, but the method was sensitive to writing distortions and segmentation quality [50].

Ghosh et al. proposed a comprehensive off-line handwritten Bangla word recognition scheme with the feature selection and using Multi-objective Memetic Algorithm, called MA-Compatible model. It used wrapper and filter approaches for finding optimal set of features. The feature extraction stage employed Statistical and Contour-Based Features (SCF), which extracted statistical shape features off the boundary contour in addition to grid-based gradient statistics from local gradient orientation information [58]. Imani et al. extracted two types of gradient features, directional and intensitybased, using a sliding vertical window that scrolls through the word image from left to right. The obtained features were codified by the Self-Organizing Vector Quantization (SOVQ) process to get a compact representation for handwritten word recognition. The sliding window approach effectively captured sequential stroke variations, although fixed window sizes could limit adaptability to

varying handwriting pattern [59]. Malakar et al. proposed an overall handwritten word recognition approach using seven low-level geometric and statistical features for each word image. These features were area, aspect ratio, number of pixels in the object, pixel density and pixel ratio (features calculated for each of the two characterizations on raw images), maximum run length and centroid and projection length important quantitative features that identify necessary structural aspects for classification. These features represented quantitative structural characteristics of handwritten words, though low-level descriptors alone could be insufficient for capturing complex handwriting variations [60]. Barua et al. proposed a holistic way for Bangla (Bengali) word recognition by dividing the image of each word into uniform grid structure and extracting SIFT like features from those grids to represent local shape and orientation properties for classification. The grid-based representation improved local feature analysis, although segmentation inconsistencies could affect descriptor stability [61].

Shruthi et al. extracted various features including density-based, long-run length and structural descriptors to represent the handwritten English word images for recognition. Long-run features captured continuous stroke distributions, while structural descriptors represented geometric handwriting properties. The method depended on accurate preprocessing and segmentation quality [62]. Aouadi et al. introduced an end-to-end model for Arabic handwritten words recognition. The approach included extracting connected components or touching characters using Ouwayed and Belaid's horizontal (left-to-right) segmentation method and curve convexity analysis for vertical (top-down) segmentation. Following the extraction of touching lines and segmentation of text lines, word segmentation was carried out, then structural features were derived from each segmented word for recognition. Curve convexity analysis identified character boundaries using contour curvature information, although complex character overlap could affect segmentation accuracy [63]. Das et al. proposed the use of the Harmony Search (HS) algorithm for feature selection to reduce dimensionality of the feature set in a holistic handwritten Bangla word recognition framework. This optimizational method was used to select the

most discriminative features so as to improve the performance and recognition efficiency [64].

Paneri et al. proposed a method for identifying Gujarati words using HOG feature extraction technique to capture specific shape and texture information [65]. Gunter et al. utilized forward and backward floating search algorithms to detect the best subsets for classification of handwritten words. These iterative search strategies dynamically add and remove features to maximize classification performance, although the search process becomes computationally expensive for high-dimensional feature spaces [66].

2.4.2 Deep Learning based Feature Extraction Techniques

An obvious trend when deep learning was introduced to computer vision, is that feature extraction changed from hand designed methods to data dependent techniques. Dutta et al., Poznanski et al., Krishnan et al., Dutta et al., and Ukil et al. employed CNNs to obtain features of handwritten word images without human input. [52] [67] [68] [69] [70]. In CNN architectures, convolution operations are performed using learnable kernels that slide across the input image to generate feature maps as shown in Eq. (2.10).

$$A_k^{(l)} = \sigma\left(W_k^{(l)} * X^{(l-1)} + b_k^{(l)}\right) \tag{2.10}$$

Where $A_k^{(l)}$ is the $k-th$ feature map at layer , $W_k^{(l)}$ is the $k-th$ learnable convolutional kernel at layer l , $X^{(l-1)}$ is the input feature map from the previous layer, $b_k^{(l)}$ is the bias term, * denote the convolutional operation and $\sigma_a\,(.)$ is the non-linear activation function. Jino et al. introduced an off-line handwritten Malayalam word recognition system using deep learning. A deep architecture is composed of transfer learning, a base CNN as a feature extractor and Support Vector Machine (SVM) classifier as a recogniser. They successfully tackled those problems resulted from complex character shapes and various handwriting styles in Malayalam. Their findings showed the effectiveness of deep architectures in improving recognition performance and thus it made an important contribution to

Indic script Handwritten recognition. However, the model performance depended on the quality of transferred features and the availability of sufficient training samples. [71].

Bluche et al. made a mixture of CNN and sliding window for feature extraction step. In this technique a window of pre-determined fixed-size of width w slides across the word image of total width W and with stride s, producing $T = \lfloor (W - w)/s \rfloor + 1$ local patches. The CNN can be interpreted as computing local features at every window position. This operation essentially transforms the original two-dimensional image into a sequence of discriminative feature vectors, which is suitable for recognition. However, the rigid pre-determined window size fails to adapt to variable character widths inherent in cursive handwriting, and the discrete stride introduces boundary misalignment at character junctions, limiting the discriminability of extracted features [15]. Wicht et al. employed the Convolutional Deep Belief Network (CDBN) trained under stacked two layers of Convolutional Restricted Boltzmann Machines (CRBMs) followed by max-pooling layer for feature extraction. Despite its hierarchy the network has been pre-trained via contrastive divergence to model word images in handwritten Arabic. Despite its hierarchical unsupervised pre-training capability for Arabic word images, the CD algorithm computes a biased gradient estimate by truncating the Markov chain, meaning the model optimises a generative reconstruction objective rather than a discriminative recognition loss. Furthermore, greedy layer-wise training prevents inter-layer feature co-adaptation, limiting the overall representational quality of the learned features [55]. Adak et al. used a CNN which was derived from the LeNet-5 network structure transited three convolutional layers to extract features from images. Training was done using Stochastic Gradient Descent (SGD) with z-scoring as a preprocessing step to normalize the feature representations. Nevertheless, the shallow three-layer architecture inherited from LeNet-5 offers limited representational capacity for the complex ligatures and contextual letter forms of handwritten script, while vanilla SGD without adaptive learning rate mechanisms remains susceptible to slow convergence and oscillation, and the global z-score

statistics computed over the training set introduce normalisation mismatch under distributional shift at test time [72].

Gupta et al. used a hybrid feature extraction method based on the Arnold transformation, orientation-based curvelet features and deep CNNs for obtaining discriminative representations of handwritten word images. The Arnold transformation applies a periodic scrambling to the image pixel coordinates providing geometric invariance prior to feature extraction. Curvelet decomposition then captures multi-scale directional features across orientation subbands, which are concatenated with deep CNN activations to form the final discriminative representation. However, the dependency on three heterogeneous feature sources increases pipeline complexity and introduces sensitivity to the Arnold iteration count, which must be manually tuned and lacks a principled selection criterion [73]. Elleuch et al. proposed strong feature extraction techniques based on DBNs and CDBNs that were trained on unsupervised data. The proposed model was tested on the HACDB and IFN/ENIT datasets for handwritten character and words classification, respectively. Despite competitive results on HACDB and IFN/ENIT datasets, the unsupervised pre-training objective remains generative rather than discriminative, and the greedy layer-wise training strategy prevents joint optimisation across layers, limiting the model's capacity to learn task-specific hierarchical representations [74]. Amrouch et al. compared two feature extraction techniques, one learned automatically by CNNs and the other handcrafted statistical features combined with structural features. Although handcrafted features offer interpretability and robustness under limited data, they rely on domain-specific engineering assumptions that fail to generalise across writing styles, whereas CNN features, while more adaptive, require substantially larger labelled datasets to avoid overfitting [75].

Wu et al. employed the PE-ResNets-BLSTM with 101 layer Residual Network (ResNet-101) for discriminative feature generation from handwritten word images. The design allows for deep feature learning, keeping computational efficiency by the use of residual connections and bottleneck blocks consisting of 1×1 and 3×3

convolutions. To retain the order of characters, we concatenate learnable position embeddings with ResNet outputs. Feature enriching spatial with context sequential for recognition better [56].

2.5 RECOGNITION TECHNIQUES OF HANDWRITTEN WORD IMAGES

Handwritten word image recognition is the core module to be analysed in offline handwriting recognition which is inflicted by different writing style, character superposition and script complexity. The early techniques of handwriting recognition were based on classical methods (such as template matching) which could work well only for constrained and synchronized data. ML-based models like SVM, Hidden Markov Model (HMM) and Multi-Layer Perceptron (MLP) enhanced the accuracy by employing handcrafted features but required pre-processing steps are more. Advances in deep learning have introduced CNNs, RNNs, and Transformers, enabling end-to-end learning and offering better performance in recognition task. This section outlines recognition techniques across classical, machine learning, and deep learning paradigms.

2.5.1 Classical or Heuristic Recognition Methods

Classical approaches to handwritten word recognition primarily focused on rule-based feature engineering and statistical modelling. Early studies largely centred on character-level segmentation and recognition. Kim and Govindaraju applied dynamic programming to align input word segments with lexicon entries in two stages: distance computation between segments and character templates, followed by optimal path alignment as shown in Eq. (2.11).

$$D(i,j) = min_k\left[D(i-1,k) + d\left(s_i, t_j\right)\right] \tag{2.11}$$

Where $D(i,j)$ is the cumulative alignment cost, s_i is the i-th input segment, t_j is the j-th character template, and $d(.,.)$ is the local distance measure. Based on training data, a variable duration model allows 1 to 4 segments per character. It reducing errors and limiting the search space. Early rejection of invalid lexicon entries was performed using segment-to-character constraints to enhance recognition efficiency [34].

Singh et al. proposed a template matching approach for handwritten word recognition, utilising the correlation coefficient as a similarity measure to identify legal amount words in bank cheques. The method was evaluated on a dataset comprising 61 distinct word classes corresponding to all legal amount terms used in Indian bank cheques and achieved a recognition accuracy of 76.4% [38]. Malakar et al. employed a method for Bangla word recognition using Genetic Algorithm (GA) framework. GA performs optimisation by evolving a population of candidate solutions through selection, crossover, and mutation operators governed by a fitness function $f(x)$ that maximises recognition accuracy. In each generation$g$, the fittest individuals are selected via tournament or roulette-wheel selection, and offspring are produced through single-point crossover and bit-flip mutation with probability p_m, iteratively converging toward an optimal feature or classifier configuration. The method was evaluated on 12,000 handwritten word images spanning 80 Bengali city names. However, GA-based optimisation is computationally expensive due to the repeated fitness evaluations across generations, and convergence to the global optimum is not guaranteed as the algorithm remains susceptible to premature convergence when population diversity diminishes prematurely [42].

2.5.2 Machine Learning based Recognition Methods

Dasgupta et al. employed a multi-class SVM for handwritten word recognition and dimensionality reduction of the extracted features with PCA. PCA projects the high-dimensional feature space into a lower-dimensional subspace by retaining the top k eigenvectors of the covariance matrix, maximising retained variance while discarding redundant dimensions. The reduced features are then passed to a multi-class SVM, which extends the binary maximum-margin hyperplane to multi-class classification through one-vs-one or one-vs-rest decomposition strategies. The proposed approach achieved an accuracy of 87.19% on the CENPARMI dataset, which contains English handwritten legal amount words [35]. Jayech et al. proposed a hierarchical framework for Arabic word recognition using Dynamic Hierarchical Bayesian Network (DHBN) structured into three layers. The top layer represents character classes, the intermediate layer models frame sets corresponding to sub-character components, and the bottom layer captures the observation nodes. The

model was tested on a sub-set of the IFN/ENIT benchmark data set to attain an average recognition accuracy of around 82% [36].

Roy et al. presented a hybrid word classification mechanism based on HMM and SVM. The word image was divided into top, center, and bottom regions. The middle zone was detected by HMM, the lower and upper zones were extracted through Connected Component (CC) analysis and identified by SVM. Experiment on handwritten word datasets in Bangla and Devanagari scripts. There are 17,091 and 16,128 word images in these two datasets. Words of length 4 were the most commonly occurring in both data-sets. The model performed at recognition rates of 92.9% and 94.51 % for Bangla and Devanagari, respectively [37]. Bhowmik et al. applied MLP and SVM for the recognition of handwritten words, in which their experimental results showed that, when training with same features extracted from a set of handwriting pattern samples, SVM was better than MLP in terms of recognition performance [32].

Kadhm et al. proposed a multiclass SVM with kernel functions such as linear, polynomial, Radial Basis Function (RBF), sigmoid. The polynomial kernel performed better with 96.317% [39]. Hegadi et al. presented a comparative study of the SVM and K -Means classifiers, used for handwritten word recognition task on their own developed dataset which comprise of 30 distinct names and 174 taluk names from Karnataka state. Experimental results demonstrated that the SVM classifier had better performance than K-Means, with the recognition accuracy of 85% [40]. Al-Nuzaili et al. utilized Extreme Learning Machine (ELM) classifiers for handwritten word recognition, based on a majority voting strategy to identify a final decision from the three word hypotheses obtained independently. Experiments were conducted on the AHDB dataset which contains 47 most common arabic words used in cheque writing from 100 writers. Under three-fold cross-validation, the recognition accuracy of 64.36% was achieved by proposed system [41].

Stauffer et al. used graph-based projection for dimensionality reduction and KNN based classifier. The graph projection embeds high-dimensional feature vectors into a lower-dimensional space by preserving pairwise relational structure through

adjacency matrix decomposition, where the reduced representation retains the most discriminative graph-spectral components. Classification is then performed by KNN, assigning a test sample to the majority class among its k nearest neighbours under a distance metric. The model was tested on the George Washington (GW) database, where we obtained a recognition rate of 82%, which indicates that the proposed methodology as an efficient technique [43]. Zaghdoudi et al. used the SVM and Fuzzy K-Nearest Neighbor (FKNN) classifiers to create four classifier combinations. FKNN extends standard KNN by assigning fuzzy membership grades. Different fusion approaches were investigated for combination of the decisions from the classifiers, where we found that Bayesian combination method achieved best results with a recognition accuracy of 98.34% [44]. Sawant et al. developed a classification model implemented by feedforward neural network algorithm, reinforced with evolutionary computation for optimisation. The proposed system is tested on a dataset of 200 handwritten samples of Marathi character and achieved a recognition accuracy of 94% [45].

Patel et al. used KNN algorithm for classification, where they applied the Euclidean distance metric for similarity computation. A study done on Karnataka districts name to test the effectiveness of text prioritization also reported 90% accuracy with 30 handwritten samples (15 writers each produced 20 different district-names) [46]. Sushma et al. used a left-to-right HMM for classification, which is the Bakis model. The resampling(second stage) was performed using the RANSAC algorithm for improving robustness against outliers, and the matching(first stage) was carried out by Brute Force. The method reached to a recognition rate of 75%, reflects the limitations of this pipeline, where Brute Force matching scales quadratically with dataset size, and the Bakis model's strict left-to-right constraint cannot accommodate backward strokes or overlapping characters common in cursive handwriting [47]. Thomas et al. proposed a lexicon-free approach for word spotting based on deep HMMs for line-level recognition. The deep HMM replaces conventional Gaussian emission models with neural network-estimated state posteriors converted to scaled likelihoods via Bayes' rule [50].

Ghosh et al. used an MLP classifier for recognition of handwritten Bangla words with experiments performed on a dataset containing 7,500-word images in 50 classes. The data set consisted of widely used city names from Bengal, each with 150 samples, written by more than 250 writers. The system attained a recognition rate of 93% with combination of feature vectors [58]. Imani et al. used HMMs to recognize handwritten Persian words by using training and testing experiments on the FARSA data base, consisting of Farsi 30,000 word images from 300 frequent words used as formal ones. By employing the directional gradient features for representation, a recognition rate of 69.07% was achieved using the system [59]. Malakar et al. employed several classification methods such as MLP, Sequential Minimal Optimization (SMO), Logistic Regression Model (LRM) and Naive Bayes (NB) for offline handwritten Hindi word recognition. With respect to the dataset used, it consisted of 4,620 word-images corresponding to names of Indian state and union territory capitals. The dataset consists of 33 classes with 140 samples per class. The MLP attained the best recognition rate of 96.82% compared to the considered classifiers with a 7-fold cross-validation scheme [60].

Barua et al. used SMO optimisation algorithm for classification and experimented using 10,000 Bangla handwritten word images containing 20 WB city names written by 500 individuals. The recognition rate of the proposed system was 90.65% with 5-fold cross validation [61]. Shruthi et al. introduced SVM based classifier for Handwritten word recognition and experimented with dataset containing English representations of Karnataka district names written by 200 people. The resulting recognition rate was 88.13% [62]. Aouadi et al. proposed an HMM of a right-to-left topology just tailored to Arabic handwritten word recognition. The system was tested on historical Arabic manuscripts and the IFN/ENIT benchmark dataset, which provided a recognition accuracy of 87% [63].

Das et al. used a MLP classifier for Bangla word identification. For experiments, a dataset of 1,020-word images divided into 20 classes was employed and each class had 51 samples. The recognition accuracy of the proposed HMM was 90.29% [64]. Paneri et al. performed comparative study of classification models, SVM and K-

Nearest Neighbor (KNN) for handwritten Gujarati word recognition. They worked on the self-built dataset with 2,700 samples of handwritten Gujarati city names. An SVM classifier reported 85.87% recognition accuracy on this dataset [65].

Gunter et al. proposed a ten HMM classifiers ensemble by using the random subspace method to build multiple classifier system based on two objective functions. Experimental results using an English sentence dataset and the IAM dataset were commaed, with an overall recognition accuracy of 71.58% [66]. Ukil et al. used MLP classifier: had been done for recognizing the handwritten words on PHD_Indic_11 dataset which has 11000 word samples covering 11 Indic scripts. The proposed model obtained an accuracy of 94.73% [70]. Gupta et al. proposed a word recognition system using three SVM classifiers. The results generated by the individual SVM classifiers were combined to improve the prediction performance where a combiner was employed. The assessment is carried out over three benchmark datasets: CENPARMI, IAM, ISIHWD. The corresponding recognition accuracy reached 95.23%, 95.07% and 97.16% [73].

Amrouch et al. used a reference HMM for automatic written word recognition and their approach was tested on the IFN/ENIT data set. The performance of two feature extraction strategies were compared in the study and an observation has been made that HMM with CNN based features provided an accuracy of 88.95% against 87.93% when handcrafted statistical and structural features are used along with HMM [75].

2.5.3 Deep Learning based Recognition Techniques

With deep learning methods, handwritten word recognition has been enhanced with the help of automatic feature extraction directly from input images. CNNs, Bidirectional Long Short-Term Memory (BLSTM), Transformers etc., are capable of automatically extracting hierarchical and contextual features which makes them useful for dealing with complex scripts and large-scale corpora. Dutta et al. utilised BLSTM network to model the sequences, which CTC loss was employed for forcing predicted label sequence to synchronize withe input features. The CTC objective function marginalises over all valid alignment paths mapping the input

sequence to the target label sequence. The performance of the model was tested on three datasets IAM, RIMES and George Washington (GW) to evaluate its efficiency in handwritten word recognition. However, CTC assumes conditional independence between output labels at each timestep given the input, which fails to capture linguistic dependencies between successive characters, and the model's performance remains sensitive to the quality of the input feature sequence fed into the BLSTM, meaning errors in the upstream feature extraction stage propagate directly into the CTC decoding process. [52].

Bluche et al. recognized words by open bigram and incorporated BLSTM-RNNs with CTC in a joint training manner. Open bigrams encode co-occurrence relationships between character pairs at variable distances within a word, providing a richer linguistic prior than unigram or contiguous bigram models. The joint training framework simultaneously optimises the acoustic-visual feature model and the linguistic bigram constraints within the CTC objective, allowing the model to leverage long-range character dependencies during sequence decoding. However, open bigram representations grow combinatorially with vocabulary size, increasing the dimensionality of the linguistic attribute space, and the joint training strategy introduces optimisation complexity where the CTC loss and bigram objectives may compete, requiring careful balancing to prevent one objective from dominating convergence [15]. Dutta et al. proposed a hybrid CNN-RNN architecture incorporating a STN, residual convolutional blocks, and BLSTM layers, with CTC loss employed for label prediction. In this, STN applies a learnable geometric transformation to spatially normalise input word images prior to feature extraction, reducing sensitivity to writing style variations. The CNN component is responsible for spatial feature extraction in this architecture, while the RNN component handles sequence-to-sequence transcription. To pre-train the model, 1 million words were rendered in both Bangla and Devanagari scripts by using fifty unique fonts for each script. The network was subsequently fine-tuned on real data using the training and validation sets of a publicly available benchmark Indic Word Database (RoyDB) comprising 17,091 binarized Bengali word images and 16,128 grayscale word images collected from 60 writers [69].

Poznanski et al. addressed this problem by proposing a CNN with multiple parallel fully connected layers to estimate the n-gram frequency of handwritten word images. The system uses attribute-based encoding where an input word image is described by a number of n-grams that are computed between the entire unintended word and its subwords. To alleviate attribute dependencies, Canonical Correlation Analysis (CCA) was peformed. These properties, referred to as Pyramidal Histogram of Characters (PHOC), generate a distinct feature-vector for each word in the dictionary. The CNN was trained to map input images onto such attribute vectors, by sharing intermediate tasks between parallel layers. The last layer's outputs were normalized to probability distributions using sigmoid activation. The aggregated sigmoid cross-entropy loss was used as training objective, Stochastic Gradient Descent (SGD) optimizer with a step-by-step training strategy for improved performance. The effectiveness of the proposed approach was tested on IAM, RIMES, and IFN/ENIT datasets with handwritten English, French, and Arabic words [67].

Krishnan et al. introduced HWNetv2 model for a compact representation of word images with feature embedding and end-to-end attribute embedding. This framework jointly learned complementary features for image and text embedding based on synthetic images and textual representations. We used a CRNN model for word recognition and incorporated to it, a STN module in order to address the geometric distortion present in the input images. Features maps obtained from CNN were further fed into the stacked BLSTM layer and processed sequential feature frames in order to produce probability distributions over class labels. The target label sequence was decoded by the CTC layer from these probabilities. They evaluated IAM and GW handwritten datasets getting recognition accuracies of 95.09% (IAM) and 98.98% (GW), respectively, on query-by-string retrieval tasks [68].

Adak et al. applied a RNN model for sequences, and used a bidirectional structure to sense the context. The network structure was formed by the input and output layers, which are linked by two LSTM layers with 128-memory units and 32-

memory units. A CTC layer was added at end to enable sequence labeling without reliance on per-segmented data. The model assessment was performed with some datasets, among them is this new dataset NewISIdb: HwC. The dataset consist 212,300 handwritten Bengali characters, and RoyDB has 17,091 Bengali word images. Moreover, we tested our method on three in-house datasets which consist of 47,200, 30,600 and 29,750 word images. The recognition accuracy was 86.96% with the system proposed [72].

Oprean et al. presented a recognition approach that combines static and dynamic dictionaries in recognition of handwritten words. The sequences of word images are classified into Anchor Words (AW) that corresponds to the reliable In-vocabulary (IV) words and Non-Anchor Words (NAW), Out-of-Vocabulary (OOV) or less reliable IV terms. Trained initially on the RIMES dataset (approximately 5,000 unique words) and tuned with a set of about 1,600 of them for validation, the recognition model is based on a BLSTM network implemented in the sliding window approach. In NAW images, a dynamic dictionary is constructed using the web-related words while after identification, the NAWs are shifted into AWs. This procedure is repeated until all NAWs are removed [76].

Kang et al. applied a standard attention-based sequence-to-sequence model called Convolve, Attend and Spell to handwritten word recognition. The model is composed of an encoder, attention mechanism, and decoder and was tested on the IAM dataset which has 47,981 training words, 7554 validation words nd 20,305 test words. The encoder combines a VGG-19-BN CNN and multi-layered Bidirectional Gated Recurrent Unit (BiGRU) for spatial and context feature extraction, respectively. Two attentions, content-based and location-based attention mechanisms are investigated, the decoder is realized as a unidirection style multi-layer GRU network which produces the output sequence character by character. The model achieved the best performance when local-oriented attention with label smoothing regularization was used [77].

Ghanmi et al. proposed a handwritten Arabic word recognition system based on two connected HMMs in a Deep Belief Network (DBN). The DBN replaces

conventional Gaussian mixture emission models with deep generative representations learned through stacked RBM layers. The DBN emission probabilities are estimated by pre-training each RBM layer via Contrastive Divergence and converting the resulting posteriors to scaled likelihoods, which are subsequently integrated into the HMM framework. The system performance was tested on the IFN/ENIT database by considering 12 selected classes corresponding to Tunisian city and village names. For each class, a specific DBN model with the same architecture and different parameters was used. Models were trained with the Expectation-Maximization (EM) algorithm. The system obtained a recognition accuracy of 72.5% when testing on test data [78].

Chen et al. presented a handwritten English word recognition method by combining deep learning and Conditional Random Fields (CRFs) within the framework of deep CRF. Unsupervised pre-training with Restricted Boltzmann Machines (RBMs) is used, while perceptron-based updates are followed by stochastic gradient descent to find the final model parameters. We observed that the model outperformed state of art methods on the OCR and ICDAR2003 word recognition datasets [79].

Kumar et al. developed a handwritten word recognition system for the Devanagari script, and addressed various segmentation issues during recognition. In this system, the word image is separated into upper, middle and lower zones. For the upper and lower zones, we used a single-stage MLP classifier for character recognition and, in the case of characters in the middle zone, we employed a three-stage MLP classification system. The system has been tested with 3,500 samples belong to 200 writers of Devanagiri script and reported recognization accuracies for two character words as 80.8% and six character word recognition is reported by an accuracy rate of 72.0% [80].

Krishnan et al. proposed HWNet, a deep CNN framework for holistic handwritten word recognition and spotting, where the network learns a direct mapping from word image pixels to discriminative embedding vectors through successive convolution, batch normalisation, and pooling operations without explicit segmentation. The model was first pre-trained on a 1 million sample sub-set of the

IIIT-HWS artificial handwritten word image dataset, and further fine-tuned with real-world examples to generalize over writers natural handwriting style. Quantitatively, the model obtained word-spotting recognition accuracy of 91.58%, and a word error rate of 6.69% on transciription recognition [81].

Stuner et al. developed a lexicon-free HWR approach focusing on a cascaded BLSTM structure together with lexicon verification scheme. This design was intended to speed up computations, make languages more independent of their lexicon size and rejection handling straighter forward. The architecture consisted of three cascaded BLSTM networks and tested on the RIMES dataset. The recognition accuracy for the which model was 90.63% [82].

Elleuch et al. employed two deep learning achievements for recognizing Arabic script; DBNN, and CNN. The DBNN was trained with a two-stage strategy: an unsupervised learning for pre-training and supervised learning for fine-tuning, while the CNN model utilized convolutional filters combined sequentially with subsampling layers aiming at reducing feature dimensionality followed by fully connected layers. The models were tested on the HACDB dataset, that consisted of 6,600 samples of handwritten characters belonging to 50 writers and augmented with elastic distortions. The DBNN was demonstrated to outperform the CNN in the experimental results [83].

Roy et al. proposed a handwritten text recognition system based on DBNN and tandem HMM. It is based on the column-wise handcrafted features of binary word images which can represent spatial information well. The performance of the model is evaluated on benchmarks (RIMES, IFN/INIT) and it outperforms traditional tandem models (MLP-HMM) has been shown. [84].

Almodfer et al. applied a handwriting Arabic word recognition technique based on Multi-Column Deep Neural Network (MCDNN) architecture. This network is a three parallel DCNN, trained separately by the same labelled data input conditioned using different normalization approaches to enhance robustness. The three outputs from each network were averaged to generate the ultimate driving risk score. The approach was tested on IFN/ENIT dataset and obtained a recognition error rate of 8.5% [85]. Amrouch et al. deployed a hybrid recognition method using CNNs for feature extraction and HMMs for sequential analysis. For experimentation, we used

IFN/ENIT dataset and found that it performed better than the baseline HMM method with 88.95% recognition accuracy [86]. Roy et al. employed the HWordNet model, a holistic CNN architecture for handwritten word recognition. The model uses word images rather than character level segmentation and has been shown to effectively capture local and global contextual feature [87].

Wu et al. utilized a bi-directional LSTMs to run over frame-level features in both forward and backward directions, obtaining context-based representations at different time. These features were classified by fully connected layers paved by a softmax output. To address this issue, an N-gram based frequency profiling method was explicitly used to model contextual character dependencies and also a lexicon-based post-processing module to refine the prediction which lowered Character Error Rate (CER) down to as low as 2.78%. Experimental analysis on the Esposalles and RIMES datasets resulted in the performance boosted significantly: namely, for Esposalles from 70.18% to 91.97% [56]. Malakar et al. employed word recognition model with a lottery ticket hypothesis on pruning CNNs. The framework increases the model compression rate with competitive recognition accuracy. The work provided a new state-of-art result on CMATERdb2. 1. 2 [88].

Li et al. proposed TrOCR, an all-transformer-based architecture for hand-written and print text recognition. Different from traditional OCR systems, which has CNN for image feature extraction and RNN or Long Short-Term Memory (LSTM) for sequence modeling, TrOCR integrates both image encoding and text decoding into a unified transformer framework. The visual information of the input image is processed by a ViT, obtaining a sequence of patch embeddings. On the other hand, a text transformer decoder produces the target word-piece sequence. This architecture dispenses with the necessity of recurrent connections, and hence allows parallelism during learning. The encoder and decoder components of this model rely on strong pretrained transformer architectures. In general, TrOCR is a big step toward attention-based end-to-end frameworks for handwritten word recognition, showing the progress of transformer architectures in replacing traditional CNN–RNN models [12].

Despite the substantial progress demonstrated across the reviewed literature, several persistent limitations remain inadequately addressed. Early statistical and

handcrafted approaches including HMMs, SVMs, and KNN classifiers lack generalisation across diverse writing styles and degraded document conditions, while deep learning methods such as CNNs and hybrid CNN-RNN architectures, though more powerful, remain sensitive to domain shift and are constrained by the local receptive fields of convolutional operations, limiting their capacity to capture global spatial dependencies within word images. Although transformer-based models address some of these limitations through global attention mechanisms, challenges related to computational complexity and fine-grained stroke representation persist. These shortcomings collectively highlight the need for architectures that combine efficient global feature learning with robust attention-driven recognition.

2.6 DOCUMENT RETRIEVAL APPROACHES

With the rapid development of Handwritten Text Recognition (HTR), handwritten document retrieval has also witnessed major advances. Traditional retrieval pipelines start by digitizing handwritten pages and transcribe them to machine-readable text through HTR systems. The extracted text is then indexed with Information Retrieval (IR) models in each document, such as Term Frequency-Inverse Document Frequency (TF–IDF), n-gram language model or semantic topic model. This recognition-based framework combines computer vision and IR for large-scale handwritten archives to search via text queries.

One study in the literature explored the use of word spotting for accessing historical handwritten manuscripts, particularly those with degraded, stylistically inconsistent handwriting, such as the George Washington dataset. In this approach, a document is viewed as a set of word images rather than as continuous text requiring complete transcription. Similarity between word images is computed using Dynamic Time Warping (DTW) applied to multidimensional profile features extracted from each word. To organize visually related word forms, clustering techniques such as K-means and agglomerative hierarchical clustering are employed, after which a subset of clusters is manually labeled to form an index. This strategy enables retrieval based on visual patterns. Since the system relies on visual similarity rather than full text recognition [89].

Fischer et al. proposed a lexicon-independent handwritten word-spotting method based on character-level HMMs. In this, the system eliminates the need for explicit word segmentation by modelling each character with an HMM trained from transcribed line images. In the retrieval phase, we represent a query word by a sequence of character models. The probability that this sequence exists in text line is calculated and then normalized by using a filler model of general text. The technique involves pre-processing stages of binarize, normalize and sliding window feature extraction. It was shown that compared to DTW on the IAM, George Washington, and Parzival datasets, the HMM framework consistently outperformed it, especially when high variability in handwriting styles are present [90].

Rowtula et al. designed an automated marking of the answer scripts on word spotting with semantic retrieval approach. Their methodology not only partitions handwritten responses into text but also transcribes them. It then adds to the retrieval by examining LDA topic patterns, not just keyword matches. This combination of image processing, text extraction and topic modelling allows a better matching between an student's free-hand answer material and the reference. In this paper, we found that the utilization of LDA semantic representations increases significantly the performance scores in both retrieval and scoring when compared to keyword-based querying methods. The work shows us the importance to add topic-level content in handwritten document retrieval, which is crucial on some applications where the semantic aspect has an important role [91].

Tüselmann and Fink investigated the semantic processing of handwriting documents in both transcription-based and transcription-free neural document retrieval methods. Their HTR-based model combines traditional handwriting recognition with downstream NLP, which however may face error propagation. To address this, we proposed the HTR-free approach which learns semantic embeddings directly from handwritten word images without an explicit transcription. Cross-modal knowledge distillation is applied to fuse visual features and pre-trained textual embeddings to have richer semantic representation. Results on semantic word spotting, named entity recognition, and question answering tasks, with IAM or George Washington datasets were presented, demonstrating that distilled visual embeddings are competitive or even better in performance in challenging handwriting scenarios [92].

Pal et al. proposed a recognition-focused document retrieval system for handwritten documents, which is evaluated on the HW-SQuAD and BenthamQA datasets. The retrieval model fuses the baseline TF-IDF vectorization and sentence-level semantic embeddings produced by transformer-based sentence encoders. The proposed approach improves traditional retrieval by overcoming draw backs of exact keyword matching. It relied on dense vector embeddings to identify semantical similar content. This resulted in a new top-5 document retrieval accuracy of over 95% on the HW-SQuAD dataset, outperforming previous benchmarks. Tokenization, stemming and ensemble-based similarity scoring are three advanced natural language processing techniques that have been incorporated within the system to provide enhanced document ranking. And, by pairing powerful retrieval models with transformer-based reader models for answer extraction, Pal et al. achieve significant performance improvements in QA for modern handwritten complex documents [93].

Recent work in handwritten document retrieval has increasingly moved toward transformer-based language models that generate rich sentence-level embeddings. Studies such as Pal et al. [93] have shown that these models can capture subtle semantic relationships in transcribed text and therefore deliver retrieval results that go well beyond simple keyword matching. Their effectiveness is particularly evident in collections where handwriting variability and noise often limit the usefulness of exact word-based search methods.

The present study adopts a more classical information-retrieval strategy to check the feasibility of the study. Instead of transformer models, TF–IDF representations are used to convert the recognized text into weighted term vectors, and LDA is applied to group sentences according to underlying thematic structure. This approach was chosen because transformer-based NLP methods generally require large volumes of annotated training data, which are currently unavailable for Malayalam handwritten document collections. In this, TF–IDF and LDA provide reliable baselines and effective matching of sentence-level queries to content extracted from handwritten pages.

If more resources are available, the system can be scaled up to include large language models and contextual embeddings. This integration would increase the semantic richness of the handwritten content, and improve retrieval performance.

2.7 SUMMARY

The literature review section in this chapter provides details on the evolution of research in handwritten word recognition and its associated preprocessing, feature extraction, classification, and document retrieval methods. A detailed survey of publicly available offline handwritten word datasets shows that substantial progress has been made for scripts such as English, Arabic, and Bengali. At the same time, resources for Malayalam remain extremely limited. This gap underscores the need for constructing a dedicated handwritten dataset for the Malayalam script.

Preprocessing techniques were examined with emphasis on noise reduction, binarization, skew correction, contour-based segmentation, and other image enhancement procedures that influence the quality of downstream recognition. These studies demonstrate that effective preprocessing is crucial, especially for documents containing varied handwriting quality, uneven illumination, and structural inconsistencies.

The review of feature extraction methods presented two major paradigms: handcrafted descriptors and deep learning–based representations. Handcrafted methods such as HOG, LBP, and zoning have historically been used, but they lack limited discriminative power when dealing with complex Indic scripts. Deep learning–based techniques, particularly CNNs and transformer architectures, are practical approaches because they can learn hierarchical, context-aware features directly from word images. The survey also highlighted recent trends toward attention mechanisms and transformer-based models, indicating a shift toward architectures capable of capturing long-range dependencies and richer visual context.

Several recognition techniques were explored, ranging from classical heuristic systems to machine learning methods such as SVMs and KNN, and modern deep

learning models including CNN–FFNN pipelines, CNN–LSTM hybrids, and transformer-based recognizers. Prior studies show that while traditional models have been effective for constrained datasets, deep neural networks offer superior robustness against variations in handwriting style, noise, and word shape complexity. However, only a few works have attempted recognition in Malayalam, revealing significant gaps in model development for this script. The absence of large-scale publicly available Malayalam handwritten word datasets, the structural complexity of Malayalam script, and the limited exploration of transformer-based hybrid architectures indicate significant research gaps in this domain. These limitations motivated the development of the proposed Malayalam Handwritten Word Dataset (MHWD) and the TrA-MHWR recognition framework presented in this study.

The final section reviewed document retrieval approaches, including keyword spotting and topic modeling. These methods demonstrate how recognized text can be transformed into indexable units for large-scale retrieval applications. The review also shows that integrating handwriting recognition with retrieval techniques is an emerging direction, especially for digitizing administrative or legal records.

Overall, the literature review identified key challenges, including the scarcity of Malayalam datasets, the need for robust segmentation methods, and the importance of advanced feature-extraction and recognition models for Malayalam script. The review also situates the research within current trends, highlighting the relevance of transformer-based recognition models and retrieval systems as promising solutions for Malayalam handwritten document processing.

In general, the review of literature lists important challenges in this area which include scarcity of Malayalam datasets, robustness around segmentation methods and need for advanced feature-extractor and recognition models specific to the Malayalam script. The review provides new orthogonal perspective to the study, in the context of current trends emphasizing that transformer based recognition model and retrieval system are potential solutions for Malayalam handwritten document processing.

3. METHODOLOGICAL FRAMEWORK FOR DEVELOPING THE MALAYALAM HANDWRITTEN WORD DATASET (MHWD)

3.1 INTRODUCTION

The absence of a publicly available and sufficiently large handwritten word dataset for the Malayalam script hinders advancements in word-level handwritten recognition, particularly for the low-resource Malayalam language. To address this gap, the present chapter outlines the first significant contribution of this research, the construction of the MHWD. This dataset is a carefully curated offline collection of handwritten word images of FIS document aimed at supporting the development and evaluation of recognition systems for the Malayalam language. The dataset is created by taking the word samples from handwritten First Information Statements (FIS) collected from the District Crime Records Bureau (DCRB), Kozhikode, Keralam**.** These real-world documents provide diverse handwriting styles, ink characteristics, and varying degrees of document degradation.

To construct a word-level handwritten dataset, it is essential to accurately segment individual words from the collected document images. This process begins with a series of preprocessing operations to improve the quality of the scanned images. Following this, edge detection techniques are applied to facilitate the precise localization of textual components. Contour-based segmentation is then employed to extract word samples from the preprocessed document. This method handles the cursive and compound character structures typical of Malayalam script. The final lexicon used in this study was produced by directly examining the set of segmented word images obtained from the scanned FIS and manually selecting the most repeated words from the segmented words. Through this process, 321 frequently occurring words were identified and selected as the representative classes for constructing the dataset.

Additional handwritten samples corresponding to the 321 selected words were collected from individuals in an age range of 10 to 65 to achieve lexical balance and enhance variability. It helped to include people from different backgrounds and a variety of handwriting styles. Each word class was uniformly populated with 150 samples, yielding a total of 48,150 handwritten word images. The resulting dataset is balanced, diverse, and well-suited for supervised learning, serving as a foundational resource for benchmarking recognition algorithm. Algorithm 3.1 provides step-by-step procedure for the MHWD dataset creation. Figure 3.1 shows the pipeline for MHWD creation.

Algorithm 3.1: Construction of the Malayalam Handwritten Word Dataset (MHWD)

```
Input:
    D_raw: Set of handwritten FIS document images
    S_add: Supplementary handwritten samples collected from
individuals

Output:
    MHWD: Labeled dataset with 321 classes × 150 samples each

  Initialize unlabelled pool P ← ∅
  for each document image I in D_raw do
      I_p ← Apply noise preprocessing to I
      C ← Extract contours from I_p
      B ← Generate bounding boxes for word regions
      for each bounding box b in B do
          w ← Crop word region from I_p using b
          Add w to pool P
      end for
 end for

 T ← Perform manual inspection on P to list distinct word tokens
 F ← Compute frequency of each token in T
 W ← Select top 321 most frequently occurring tokens

 for each word class wᵢ in W do
      Collect additional handwritten samples from S_add
      Validate and retain only clean samples
      Ensure exactly 150 samples for class wᵢ
      Store samples in class folder wᵢ
 end for

 Construct MHWD by combining all class folders
return MHWD
```

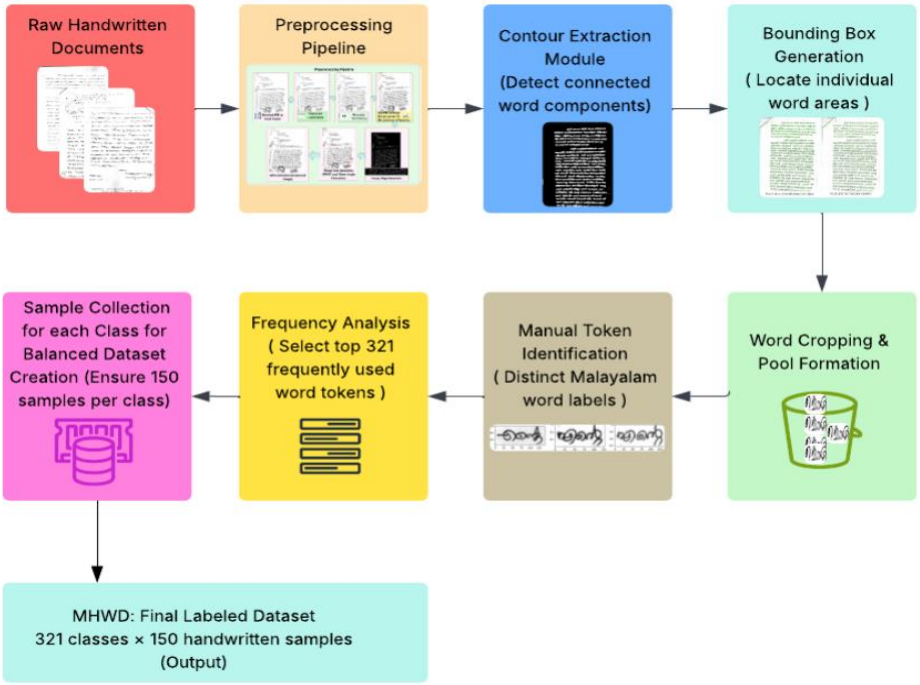

Figure 3.1 System Architecture Diagram for MHWD Creation

This chapter details each phase of the dataset creation pipeline, including document collection, preprocessing, segmentation, lexicon formulation, and additional sample acquisition to form a balanced dataset. Besides, statistical details about the word samples are presented in this chapter.

3.2 ACQUISITION OF HANDWRITTEN IMAGES FROM POLICE FIRST INFORMATION STATEMENTS (FIS)

To build a reliable and domain-relevant offline handwritten word dataset, it was necessary to identify a document source that captured real-life handwriting diversity along with operational significance. After exploring various institutional and administrative sources, the police's First Information Statement (FIS) was identified as the most suitable choice for this work. The FIS serves as an official primary legal document in which the initial account of an alleged offence is recorded based on the complainant's report. In Keralam, FIS documents continue to be manually written by the police personnel responsible for recording the complaint. So, the manual handwritten FIS documentation process helps to collect handwriting samples with considerable variation in writing styles, ink types, and formatting. Such variability

makes FIS documents particularly valuable for developing and evaluating systems aimed at handwritten word recognition.

An essential reason for choosing FIS documents is their relevance in progressing digitization efforts. Many older FIS documents remain handwritten and are now being scanned for archival and judicial purposes. Some of these records are still active cases, necessitating the preservation of accurate and readable digital versions. Additionally, recent directives have emphasized the need to prepare typewritten versions of FIS content to improve clarity and aid interpretation during legal proceedings. This creates the need for automated tools that can assist in digitising and recognising handwritten FIS documents.

To address this need and ensure practical requirements, a collection of 1,000 handwritten FIS documents was obtained from the District Crime Records Bureau (DCRB), Kozhikode. This data collection contains handwritten documents from various police personnel with unique stylistic variations. As a result, the collected FIS documents offer a realistic and domain-specific resource for developing and testing handwritten word recognition models, particularly in legal and administrative contexts. The dataset can also be used for search and document retrieval. Figure 3.2 shows sample images of handwritten FIS.

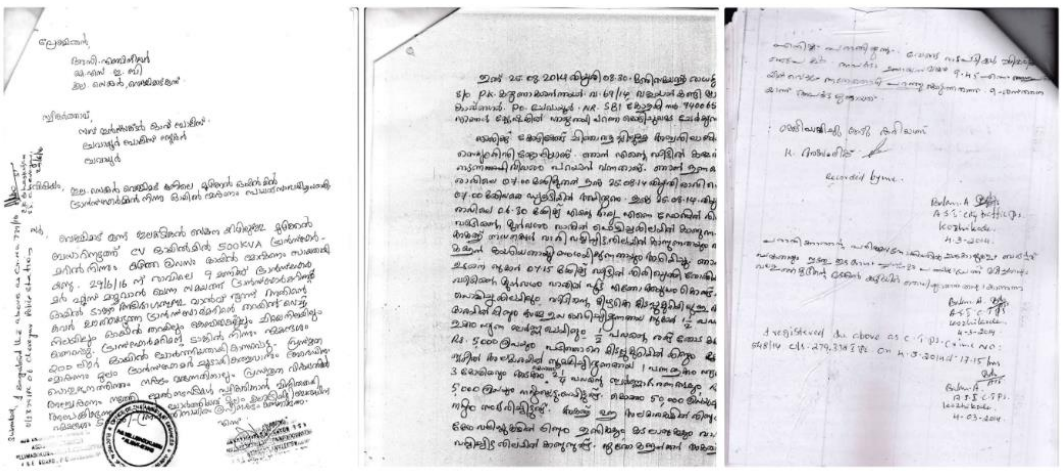

Figure 3.2 Handwritten image samples of FIS document

The handwritten FIS documents contain a rich vocabulary of legal and administrative terms, including specific Malayalam terms related to police procedures, such as സ്റ്റേഷൻ (Station), പരാതി (Complaint), and ആത്മഹത്യ (Suicide). It also contains temporal expressions (dates like 16-06-2021 and 20-05-2021), case numbers (548/14), monetary amounts, personal names, geographical locations, and detailed narrative descriptions of incidents, including theft, fraud, and domestic disputes. The statements are generally written in a free-flowing manner, with a paragraph-based organisation, and are not strictly aligned or segmented. Police write on unlined paper, the space between lines and letters changes quite a lot.

The data collection consists of scanned images of handwritten FIS documents provided by DCRB in PNG format. Explicitly, the FIS images are chosen from non-identifiable cases to satisfy legal and privacy requirements. These forms show significant amounts of handwritings variability between various officers, such as different forms of ligature usage and stroke thickness, degree of cursiveness, slant/skew angles. Official circular stamps, superimposing over the underlying Malayalam text, are observed in the documents which pose additional preprocessing hassles. Moreover, the scanned pages may exhibit uneven background discolouration, faded or bleached text parts, paper texture effects and artefacts, scanning shadows at edges of document and bleeding of ink and blot footprints which complicate further processing such as preprocessing and segmentation.

An average of 50–160 words exist on each scanned page image of FIS varying depending on narrative length and writing density. The FIS case statements not only offer real, spontaneously produced examples but also include domain specific vocabulary. This lexicon includes words regarding legal and administration with frequent usage, and the words that appeared in the incident narrative and complaints.

3.3 PREPROCESSING PIPELINE FOR ENHANCEMENT AND SKEW CORRECTION OF SCANNED HANDWRITTEN FIS DOCUMENTS

The preprocessing plays cruicial role in improving the quality of scanned handwritten document images, especially for the reliable segmentation and recognition. Many restoration artefacts are visible in historical documents like FIS records. These are noise patterns, non-uniform illumination, low contrast and corrupted background namely due to paper ageing ink spread or scanner limitation.

Geometric distortions such as skew, is prevalent in handwritten documents. This is what's known as skew, or the tilt at which lines of text are printed – often due to shaky handwriting, or having been crooked while placed against a scanner. Unaddressed, these distortions can have severe impacts on later processing steps such as word-segmentation and character recognition.

Such challenges are addressed with a preprocessing pipeline, which includes both image enhancement and geometric correction. The phases are: Conversion to Grayscale, Gaussian blur, contrast enhancement by Contrast-Limited Adaptive Histogram Equalisation (CLAHE), morphological noise reduction, edge detection by Canny and skew correction using Hough Line Transform.

3.3.1 Grayscale Conversion

The first operation applied to the input image is grayscale conversion. This step accomplished using OpenCV's cv2. imread() and the cv2. IMREAD_GRAYSCALE flag which reads the image in a single intensity channel as opposed to reading three-channel RGB by default. For an RGB image, each pixel is represented using three values of the RGB colour intensities and this representation is important for natural scene understanding; however, it is generally unnecessary and computationally expensive for document processing applications such as skew estimation segmentation and recognition. In the processing of handwritten documents the color is not of interest but rather the morphology and structure of the written text. Grayscale converting eliminates that diversity of the

data coming from three different color RGB values for each pixel and makes its value a weighted sum of red, green and blue values by the Eq. (3.1).

$$I(x,y) = 0.299 * R(x,y) + 0.587 * G(x,y) + 0.114 * B(x,y) \quad (3.1)$$

Here, $I(x,y)$ represents the grayscale intensity at pixel(x,y). $R(x,y), G(x,y)\ and\ B(x,y)$ are the red, green and blue channel intensities. Grayscale conversion can also reduce the amount of memory needed and computation for further image processing. Correspondingly, the transformation into a grayscale image allows for reduced speed and improved effectiveness through the removal of redundant ink information and by preserving spatial structure and stroke properties of handwritten notes.

3.3.2 Gaussian Smoothing for Noise Suppression

The first enhancement step is performed by a Gaussian filter [94] to reduce high frequency noise and smooth small scale variations in intensity which are typically introduced by scanning artifacts, paper texture or dirt. Gaussian smoothing is a low-pass filter that suppresses high frequency noise by the convolutions of image with the Gaussian kernel. This is particularly useful on salt-and-pepper noise or small speckles that are not part of the actual text. The Gaussian image filter is based on the 2D Gaussian distribution in Eq. (3.2),

$$G_k = \frac{1}{2\pi\sigma^2} \exp\left(-\frac{x^2 + y^2}{2\sigma^2}\right) \quad (3.2)$$

where x and y represent pixel distance from the kernel center. σ is a smoothing parameter. The input grayscale image $I(x,y)$ is smoothed using convolution operation, as shown in Eq. (3.3)

$$I_s(x,y) = I(x,y) * G(x,y) \quad (3.3)$$

Near the centre of the kernel, pixels are assigned higher weights during convolution, achieving noise reduction while maintaining important structural information. In the image enhancement step, Gaussian smoothing (OpenCV: cv2. GaussianBlur) with

a 5×5 kernel and automatically determined standard deviation (σ = 0.67). Such operation suppresses effectively high-frequency noise associated with aged scanned document and preserve useful text structural information, as recommended in foundational image processing literature. The balance in parameter choice conforms to good practices for historical document restoration in which over-smoothing may cause the destruction of delicate stroke structures.

3.3.3 Local Contrast Enhancement Using Contrast-Limited Adaptive Histogram Equalisation (CLAHE)

The image is then passed through CLAHE to improve visibility of the handwritten text where ink might fade or with an uneven background after noise removal [95]. In contrast to global histogram equalisation, which uniformly modifies the image contrast, CLAHE divides an image into a number of smaller scales (referred to as tiles) and performs local contrast enhancement in each tile separately. The histogram is clipped at a predetermined threshold, as shown in Eq. (3.4).

$$h'_k = min(h_k, C) \tag{3.4}$$

Where $h_k$the original histogram count of grey level k, and C is the clip limit to prevent over-amplification for contrast. And the clipped excessive is evenly distributed over all gray levels according to Eq. (3.5).

$$h''_k = h'_k + \frac{E}{L_g} \tag{3.5}$$

E is the clipped extra pixels, and L_g is the number of gray levels. The averaged intensity mapping is obtained by the cumulative distribution function (CDF), in Eq. (3.6).

$$I_{out} = (L-1) \times CDF(I_{in}) \tag{3.6}$$

I_{in} and I_{out} refer to the input and output pixel intensities, respectively.

Using bilinear interpolation, we interpolate between neighboring tiles to guarantee smooth updates and to prevent blocking artifacts. CLAHE was used with a 2.0 clip

limit and an 8×8 tile grid size to improve local contrast while not introducing excessive noise. This choice of parameters is according to the recommendations for document images, where the tile size should include character groupings without overlapping with light-to-dark transition in neighboring word regions. CLAHE is especially useful to improve readability of faded text: it enhances the visibility of strokes, while preserving the structural continuity of handwritten text.

3.3.4 Morphological Opening to Remove Isolated Noise Pixels

Morphological opening is applied [93] to further filter the image and remove small, isolated components after local contrast enhancement with a CLAHE. Morphological opening is the erosion followed by the dilation using the same structuring element, that is 2x2 square kernel. A mathematical formulation of the opening operation of an image by a structuring element is given in Eq. (3.7).

$$A \, o \, B = (A \ominus B) \oplus B \tag{3.7}$$

Where $\ominus$ is erosion and $\oplus$ is dilation. The purpose of this operation is to eliminate small artifacts, and isolated noisy pixels that might be still present after contrast enhancement while keeping the basic forms and size of larger significant structures like handwritten characters. Erosion operates by reducing small bright areas and thinning the text strokes while removing spurious noise. The subsequent dilation step then rescales the remaining features to their initial width, but without reintroducing the irrelevant noise. This step also contributes to creating more accurate line boundaries and sequences between the text and background, something that is crucial for reliable skew correction and subsequent segmentation work.

3.3.5 Canny Edge Extraction to Support Skew Correction

Edge detection has been a fundamental operation in digital image processing for the early stages of detecting the discontinuities in an image, which generally are associated with either surface boundaries or significant changes to perceptual content within an image. Edge detection plays a key role in the analysis of handwritten document as it allows to extract textual structure, alignment and layout

information that are essential for further preprocessing tasks. A number of classical operators are proposed for edge detection such as Sobel operator, Prewitt operator Roberts and Canny detector. Of these, the Canny edge detection method is generally considered the best [96].

The boundaries detected by the Canny operator give welldefined structural boundaries, which help increase accuracy of estimation and correction later in handwritten document image [97]. It emphasizes the linear components, such as text lines and boundary between documents, which can be utilized to estimate the skew angle. The canny algorithm works as follows: noise reduction, computation of the gradients, non-maximum suppression, double thresholding and edge tracking by hysteresis[98]. Initially, it applies Gaussian filtering to suppress image noise, followed by the computation of intensity gradients to highlight regions with sharp intensity transitions. Although earlier preprocessing included Gaussian smoothing, the Canny algorithm performs its own Gaussian filtering as part of its internal pipeline to ensure stable and noise-resistant edge detection before computing the gradient. The intensity gradient magnitude and direction are computed using equations expressed in Eqs. (3.8) and (3.9).

$$|G| = \sqrt{G_x^2 + G_y^2} \tag{3.8}$$

$$\theta_i = \tan^{-1}\left(\frac{G_y}{G_x}\right) \tag{3.9}$$

Subsequently, non-maximum suppression is employed to thin the edges by retaining only the local maxima of the gradient magnitude and it is shown in Eq. (3.10).

$$E(x,y) = \begin{cases} |G(x,y)|, if\ it\ is\ the\ local\ maximum \\ 0, \quad otherwise \end{cases} \tag{3.10}$$

The resulting image then undergoes double thresholding to distinguish between strong and weak edges. Finally performs edge tracking by hysteresis, connecting weak edges to nearby firm edges to preserve meaningful boundaries. This classifies pixels into strong, weak, or non-edge regions based on two thresholds T_H *and* T_L, as shown in Eq. (3.11).

$$E'(x,y) = \begin{cases} Strong\ Edge, & |G| \geq T_H \\ Weak\ Edge, & T_L \leq |G| < T_H \\ 0, & |G| < T_L \end{cases} \quad (3.11)$$

In the present study, edge detection is carried out using the Canny method with threshold values set at 50 (lower) and 150 (upper). A 3×3 Sobel operator (with an aperture size of 3) is employed to estimate the image gradients. It highlights prominent edges associated with handwritten text lines while suppressing background noise, thereby producing a suitable edge map for accurate skew detection using the Hough Line Transform.

3.3.6 Probabilistic Hough Line Transform for Skew Correction

Skew estimation and correction are two important preparatory blocks in the processing of a document images. As discussed by Biswas et al. [99], diverse techniques are presented for skew correction for print and handwritten documents. These measures involve projection profile based and Fourier transform-based image analysis techniques, as well as those based on Radon transform where the estimation of skew through line integrals is known to have high accuracy, among other ones listed below including simple morphological operators such as run-length encoding and smoothing. It also presents connected component based clustering methods where spatially proximal and structurally similar components are clustered to predict the dominating text orientation direction. In addition, machine learning models such as neural networks are applied to learn if skew patterns and predict the right orientation without needing manual feature engineering.

The Hough Transform (HT), in its standard, probabilistic and adaptive versions, is known to be effective for noise of boundaries or discontinuities supports on text lines. In this work, a solution is to use Progressive Probabilistic Hough Transform (PPHT) for skew correction [100]. In this way, computational efficiency is increased by adaptively determining a proper threshold for line detection and avoiding lines already having obtained points from the further detection.

The Hough Transform is a traditional method to recognize geometric shapes in images by transforming edge points from the image space to a parametric

space[101]. The classical HT assigns the polar form and then obtains lines by peak votes in an accumulator, as shown in Eq. (3.12).

$$r = x\ \cos\theta + y\ \sin\theta \tag{3.12}$$

Where, r is the perpendicular distance from line to the origin and θ is the orientation angle of the line. The PPHT technique adopts the random sampling and early termination scheme to reduce computation burden. The detected line segments are successively eliminated in the voting domain, so as to avoid redundant computation.

In this study, OpenCV HoughLinesP() function is utilized at 1 pixel distance resolution and π/180 radians of angular resolution, for accurate line orientations detection in the image. We use a minimum voting value as criterion to discard weak/irrevalent line candidates. This threshold is the minimum number of edge pixels to be supported in the Hough accumulator space for a result line to considered as detection. A large threshold will classify only strong, well-supported line segments to be line while it suppresses weak, noisy or spurious responses. A voting threshold of 100 is imposed so that only strong lines will be concerned with from the hough parameter space. Also, it is observed that other parameters such as the minimum line length of 100 pixels and the maximum gap between line segments of 10 pixels are useful to get rid of short or discontinuous detection, which properly contributes to make the skew estimation more reliable.

After detecting the line segments, the directions of lines are calculated in order to estimate the skew angle of a document. This involves using the coordinates of the endpoints $(x1, y1)$ and $(x2, y2)$ to calculate the angle of inclination relative to the horizontal axis. The inclination is determined using the arctangent function, as shown in Eq. (3.13).

$$\theta = \tan^{-1}\left((y2 - y1)/(x2 - x1)\right) \tag{3.13}$$

This is then converted to degrees for interpretation. As a result, in order to avoid confusion, and because only angles close to the vertical axis are not representative

of horizontal alignment of text lines. Thus, only line orientations in the the range −45∘ to +45∘ are taken into account. Median of the set of all angles after relief of slant is kept as an estimated document skew angle. By taking median (as opposed to mean) we can address the problem of outliers and noise, hence making the estimate more robust. The skew angle is set to zero if no candidate lines are found. This calculated angle is then used to rotate the image and undo the skew.

Once the skew angle is estimated, the document image is de-skewed by an affine transformation to ensure that collinearity and parallelism of pixels still hold. Rotation is performed by using matrix multiplication of the image coordinates with a 2D rotation matrix expressed in Eq. (3.14).

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} \cos(-\alpha) & -\sin(-\alpha) \\ \sin(-\alpha) & \cos(-\alpha) \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} \tag{3.14}$$

Where (x, y) are the original pixel coordinates, (x', y') are the transformed pixel coordinates, α is skew estimation angle and negative rotation (-α) can be used to correct for detect tilt to get an upright document. So we first calculate the centre of the image using its width and height. A 2×3 rotation matrix in OpenCV is formed using getRotationMatrix2D() method, where the input arguments are, image center, estimated skew angle (in degrees) and a scaling factor of 1.0. The resultant T matrix is then used with the warpAffine() function to rotate our image: In this operation, the region along the border introduced by rotation is filled with a constant grayscale level of 255 (indicates pure white in an 8 bit gray-level image). Linear interpolation is applied to obtain smooth pixel transitions post-transformation.

The performance of the skew correction method was tested on three sample document images having varied amount of skew. Figure 3.3 shows the sample input images with estimated skew angles along with their corresponding output corrected images.

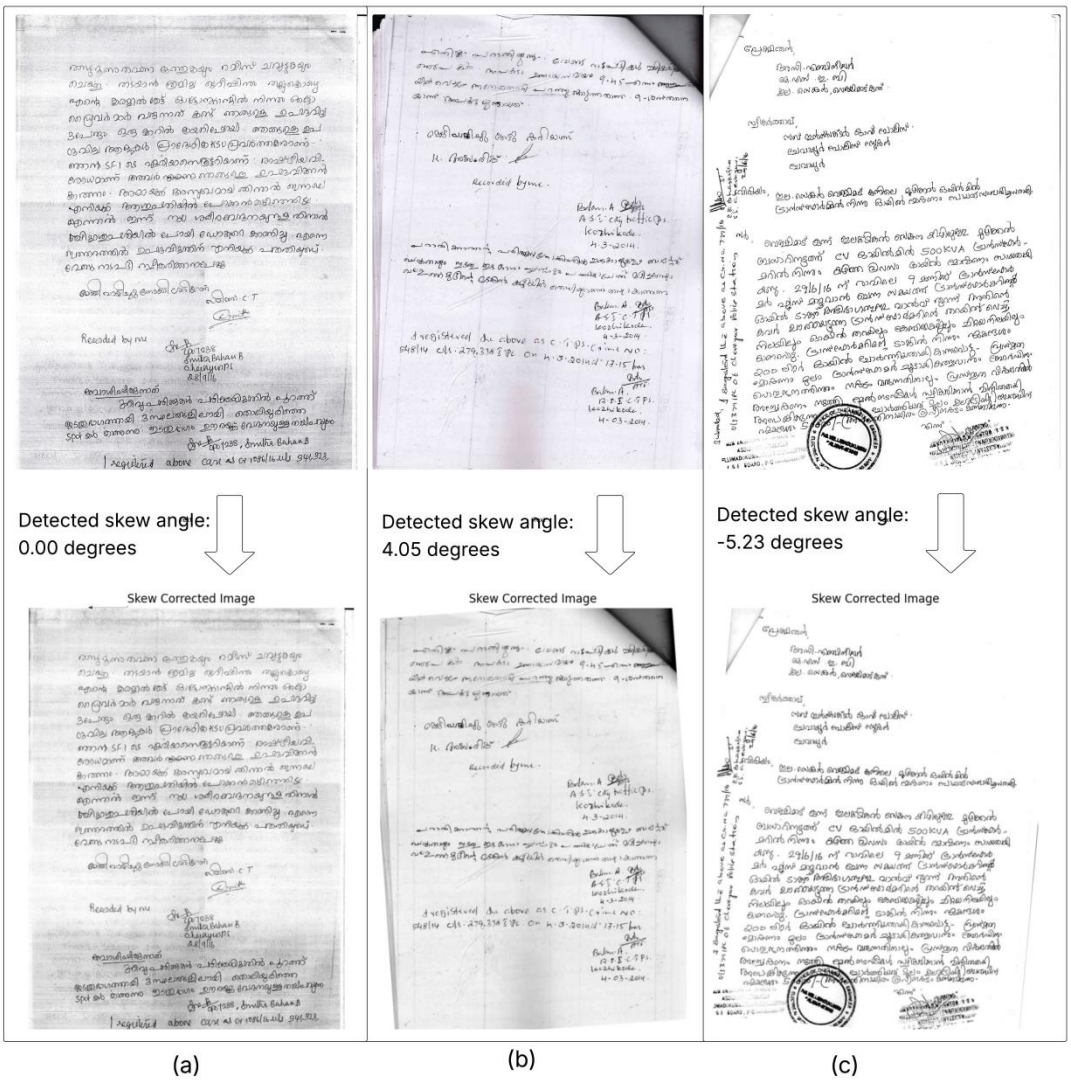

Figure 3.3 Skew Correction Results

The method of skew correction was tested by three document images with different degrees of skew. The subfigures (a), (b) and (c) in the skew correction results stand for recognized skewing angles of 0.00°, 4.05°, and −5.23°, respectively.

Figure 3.2 (a) shows a correctly aligned document where the method detected a 0.00° skew angle This observation ensures that the orientation is unchanged at all, if you don't need to correct anything. Subfigure (b) shows that the method handles even a small level of rightward skew, such as 4.05°, and corrects it to bring back horizontal text line alignment. In subfigure (c), the method detected a left skew of −5.23°. After the alignment was successfully restored, it performed an affine transformation. These are sensitive cases, so we can already judge the method's

appropriateness: they're valid for everything from well-scanned pages to documents which have "a little or moderate" skew in either direction. Through the use of median angle estimation, this method improves detection robustness by being less sensitive to anomalous or outlier lines in the text. In general, the combined use of PPHT and affine transformation provides a reliable method to deslant text for various kinds of documents. Figure 3.4 presents the pre-processing pipeline and each block results in an intermediate output.

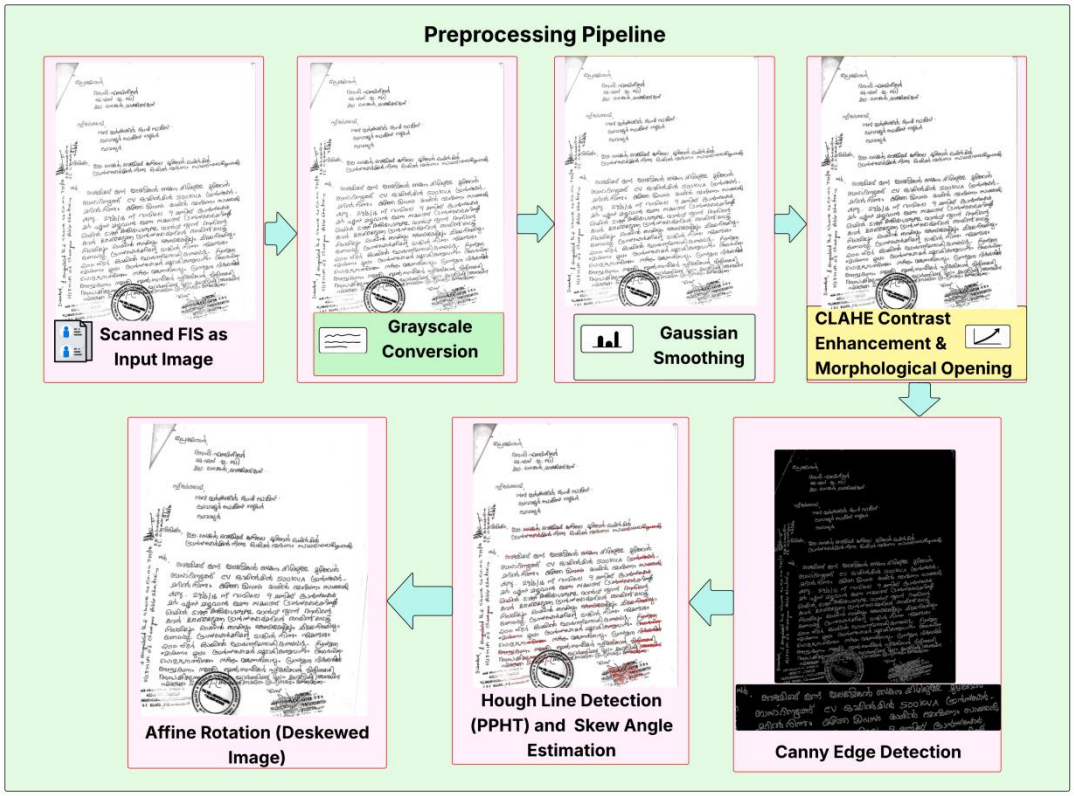

Figure 3.4 Overview of the preprocessing stages implemented for scanned handwritten documents

The block diagram shows preprocessing activities on scanned handwritten Malayalam documents. The input FIS image is at first changed to grey level to make the operation easier. Gaussian smoothing is subsequently applied to filter out noise and then CLAHE for enhancing local contrast. Morphological opening eliminates small artefacts and increases the quality of extracted text. Then, canny edge detection is applied to emphasize the structural properties of the input image and this is then utilized in the PPHT for document skew estimate. Then we deskew

(affine rotate) the image. Each stage enhances quality and alignment, which lead to more improvement for downstream segmentation and recognition.

3.4 WORD SEGMENTATION USING CONTOUR-BASED TECHNIQUES

After preprocessing and skew correction, the next task is to segment the individual words from the hand written document. This is accomplished by a contour-based method, including binarisation, morphologic operations and contour analysis for word region separation precisely.

3.4.1 Binarization of Documents using Otsu's Method

Otsu's method is a well-known global thresholding method for the generation of binary images from grayscale images, and gives relatively better results in analysis of document images and segmentation application [102]. The algorithm automatically finds a suitable threshold value to divides the pixels of the image into two classes such as foreground (the text) and background by evaluating an intensity histogram of the image.

The core principle of Otsu's method is to get a threshold that minimises the intra-class variance, which is the weighted sum of variances of the two pixel classes separated by the threshold. Equivalently, this approach maximises the inter-class variance, ensuring the most significant possible distinction between text and background regions. Otsu's method performs well on images that exhibit bimodal histograms, in which the pixel values of the foreground and background form two distinct peaks. Mathematically, for a threshold t, the intra-class variance is calculated as shown in Eq. (3.15).

$$\sigma_{\omega}^2(t) = \omega_0(t)\sigma_0^2(t) + \omega_1(t)\sigma_1^2(t) \tag{3.15}$$

where $\omega_0$and $\omega_1$are the probabilities of the two classes separated by t, and σ_0^2 and σ_1^2 are their respective variances. In practice, Otsu's method is implemented using the cv2.threshold function in OpenCV with the THRESH_OTSU flag [103]. Here, the algorithm computes the optimal threshold and applies it to the deskewed grayscale image. The THRESH_BINARY_INV flag is used to invert the output

image, making white text appear on a black background, which is advantageous for contour-based word segmentation. Otsu's method does not require manual parameter tuning and is computationally efficient, making it a standard choice for preprocessing in document image analysis. In this work, Otsu thresholding is introduced specifically at the segmentation stage because the earlier preprocessing operations (e.g., smoothing, contrast enhancement, and skew correction) require grayscale intensity information. In contrast, binarisation becomes essential only when separating foreground text from background regions.

3.4.2 Morphological Dilation for word Connectivity Improvement

Morphological dilation is an important step in the word segmentation pipeline as character spacing and writing styles may differ substantially for handwritten documents. The dilation is a morphological operation in the mathematical sense; it makes objects (in this case, white areas representing text) grow. It is an important step after binarisation, to close small gaps between characters in a word and recover fragmented region as blob for the entire word.

A rectangular structuring element was applied during the dilation operation in this process. This kernel was chosen based on an empirical study of the dataset, and considering the standard horizontal and vertical spacing present in handwritten Malayalam FIS. Symbolic dilation of a binary Image A by a structuring element B, as in Eq. (3.16).

$$A \oplus B = \{z|(B)_z \cap A \neq \emptyset\} \tag{3.16}$$

Where $(B)_z$ is the translation of B by z [104]. The dilation is computed by moving the structuring element over the input binary image. At each location, if any pixel in response to the kernel is a foreground pixel, which is white, then that pixel will be set to foreground value. Such component of mechanism extends the white areas, joining together closely spaced characters into words. The success of this operation is largely related to the shape and size of the structuring element. If the size of a kernel is too small, some characters within one word may not be connected as one

piece; if the kernel size is big then neighbour words can be joined into the same segment.

3.4.3 Contour Detection for Word Localization

Once morphological dilation has been applied, the focus shifts to locating the outlines of connected white regions in the binary image that represent individual words present in the document image. The OpenCV findContours() function is used to trace the perimeters of all distinct foreground objects systematically [103]. Contour is a closed curve joining all the continuous points along the boundary of a connected foreground region [105]. For a binary image $I(x, y)$, Pixel intensities are assigned two discrete values after thresholding: foreground (text) pixels are set to 255, and background pixels are set to 0. These values are produced automatically by the Otsu thresholding operation using OpenCV, where binary output images in 8-bit grayscale format use 255 as the maximum intensity to represent white (or black, depending on inversion). A contour C_n can therefore be mathematically expressed as the boundary between these two classes, as shown in Eq. (3.17).

$$C_n = \{(x, y) | \, I(x, y) = 255 \; and \; \exists \, (x', y') \, \epsilon \, N \, (x, y) \; : \; I(x', y') = 0\} \tag{3.17}$$

Where $N \, (x, y)$denotes the set of neighbourhood pixels. In the processing, only the outermost contours are extracted (using retrieval mode RETR_EXTERNAL) which corresponds to the main boundaries of each word candidate. CHAIN_APPROX_SIMPLE parameter reduces the number of stored points; it saves enough points to approximate each contour closely while retaining the essential shape of each contour. Each contour detected represents a unique white region in the binary image, usually representing a word. These contours are essential for further handling through procedures like computing bounding rectangles and extracting word images for latter recognition and analysis.

3.4.4 Extraction and Saving of Segmented Word Regions

Once the contours representing candidate word regions have been detected, it is necessary to place these regions in the script's natural reading order, which is left-to-

right for Malayalam. This is achieved by taking the leftmost horizontal position of every contour within the image. In particular, the x-coordinate of the top-left corner of the bounding rectangle of the contour is used as the key for sorting.

Following the sorting of contours to align with the natural order of reading, the second step is to separate each word area and store it as a single image for subsequent analysis. This process involves several scientifically sound operations that are all based on mathematical concepts from image processing.

Bounding Rectangle Computation: A bounding box is found for each detected contour. Given a contour C_n, the bounding of C_n is the minimum axis-aligned rectangle that contains all the points on C. If (x_i, y_i) are the points in the contour, then the rectangle is given by Eq. (3.18).

$$\mathrm{R} = [\min(x_i), \min(y_i), \max(x_i) - \min(x_i), \max(y_i) - \min(y_i)] \quad (3.18)$$

Where the top corner of the rectangle is at $(\min(x_i), \min(y_i))$, and its width and height are $\max(x_i) - \min(x_i)$ and $\max(y_i) - \min(y_i)$, respectively.

Spatial Filtering: To separate words from noise and minor artefacts, only the contours of the words that have the bounding rectangles that meet the minimum width and height criteria are maintained. The contour is valid if $w_{min} < w < w_{max}$ and $h_{min} < h < h_{max}$ where $w_{min}, w_{max}, h_{min}$ and h_{max} are experimentally determined thresholds. These bounds can be derived by considering the average size of word regions in the document. The lower thresholds filter out noise and small artifacts, whereas the upper thresholds avoid that large connected components in the detected bounding box, for example full text lines or multiple attached words, are falsely dealt with the facets belonging to a single word.

Region Extraction: Each legitimate bounding box is employed to crop the corresponding region from the original image. If I denotes the image matrix, and the bounding rectangle is represented as (x, y, w, h), then the corresponding word image W is given as Eq. (3.19).

$$W = I[y: y + h, x: x + w] \quad (3.19)$$

This procedure effectively separates the pixel region of that word from its spatial context. The obtained images are written to an output directory, with filenames assigned according to their horizontal coordinate. This has the advantage that the reading order is preserved and suitable for further downstream tasks.

The proposed approach is driven by the geometrical properties of the contour and bounding box to detect the words with great precision. However, by extracting according to spatial and mathematical region definitions, meaningful word segments are retained, reducing false positive detections resulting from noise or other artefacts. Figure 3.5 shows the word segmentation pipeline.

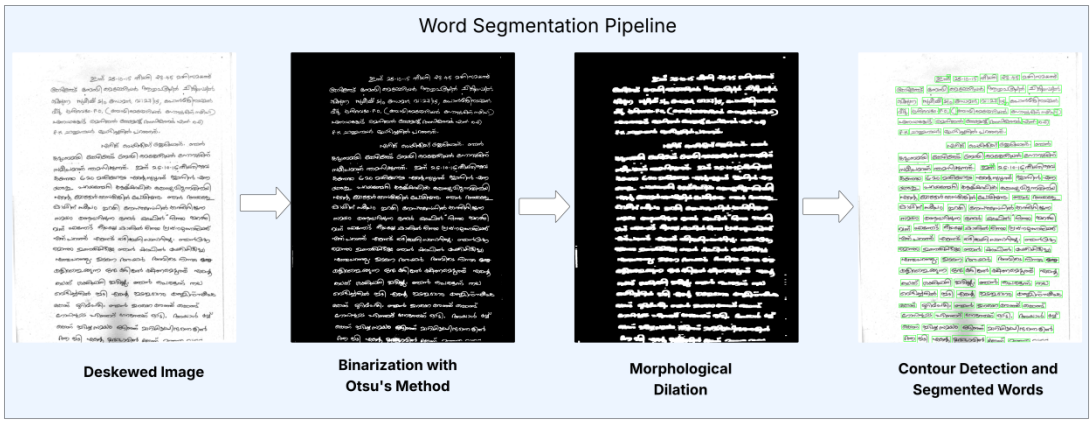

Figure 3.5 Block diagram illustrating the word segmentation pipeline using contour-based techniques with intermediate outputs at each step

The block diagram provides a concise representation of the word segmentation pipeline process. Following binarisation based on the Otsu method and morphological dilation, the last step shows the extraction of word foregrounds by drawing bounding rectangles around each word region within the processed image. The diagram provides an overview of the process, describing the transformation from the original document to the final segmented outputs, along with every intermediate output of the process.

To validate the performance of the word segmentation method, manual verification was conducted on a subset of representative scanned handwritten FIS documents. For that, three different document images were chosen, and the number of segmented word images generated by the system was matched with the actual number of words in each respective document.

The scanned image of handwritten documents was visually inspected to perform manual word counting. This count was manually determined and was taken as the ground truth for performance evaluations of the segmentation. Finally, the proposed word segmentation method is compared with this reference to evaluate its accuracy and reliability through the number of words segmented.

The performance of the segmentation process is summarized in Table 3.1. For each document, the number of obtained words (automatically segmented), the actual words (manually counted), and derived metrics such as missed words and segmentation accuracy are reported.

Table 3.1 Segmentation Performance on Sample Documents

Document	Obtained Words	Actual Words	Missed Words	Accuracy (%)
Doc 1	129	133	4	97.0
Doc 2	112	118	6	94.9
Doc 3	34	38	4	89.5
Average				93.8

The results, presented in the table, indicate that the proposed segmentation method achieved a satisfying accuracy of 93.8% on the tested document images of handwritten documents. Missed word regions were primarily to blame for most segmentation errors, frequently caused by words being very closely written or touching, especially in the case of horizontal and vertical overlaps. Thus, although this claims a limitation to disjoining components joined by dense handwritten contexts, it performed well on less dense documents as well, and the handwriting was deemed relatively accurate. Crucially, no false positives were observed, demonstrating the conservative behaviour of the technique, which is advantageous for subsequent word recognition tasks where accuracy is prioritised.

Since publicly available ground-truth datasets are not available for most low-resource languages, including Malayalam, we evaluated the effectiveness of the proposed segmentation method through a comparative study against some traditional image processing-based techniques. Some of these were Connected Component

Analysis, morphological-based segmentation, and the Projection Profile. Of these, Connected Component Analysis and morphological techniques performed better than Projection Profile-based methods. Figure 3.6 shows the results obtained after applying the segmentation techniques connected component analysis, morphological operation and the proposed method.

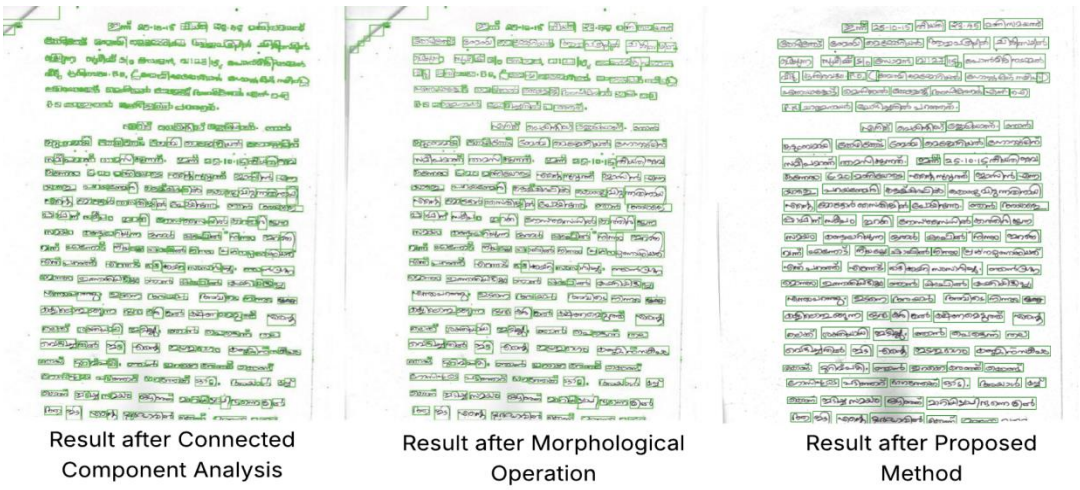

Figure 3.6 Word segmentation results using traditional methods: Connected Components, Morphological Operations, and proposed method

To assess the generalizability of the proposed segmentation method, the handwritten images of different Indian scripts taken from the PHDIndic_11 dataset are evaluated. Figure 3.7 presents examples of the segmented handwritten words in different scripts. Table 3.2 summarises the results obtained after word segmentation experiments across different scripts.

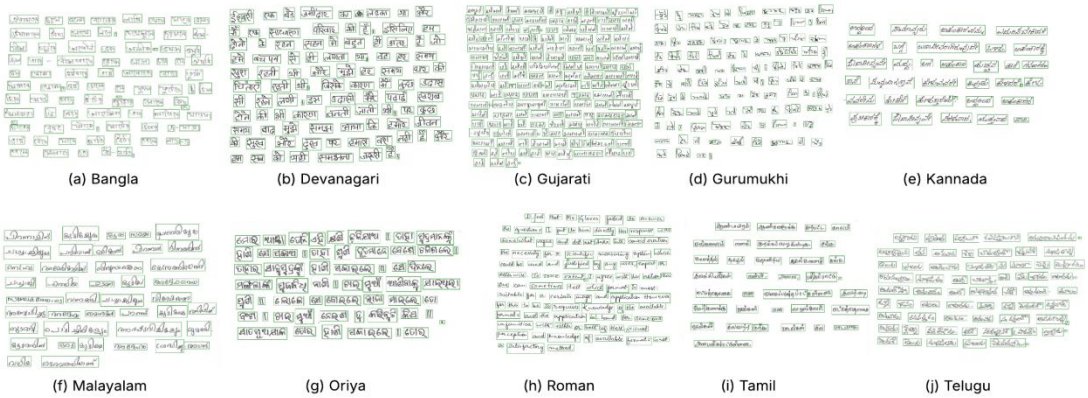

Figure 3.7 Word segmentation results on handwritten document images from 9 Indian scripts and Roman Script

Table 3.2 Word Segmentation Performance on Different Indian Scripts (Dataset: PHDIndic_11)

Script	Actual Word Count	Segmented Words (Obtained)	Accuracy (%)
Bangla	82	79	96.34%
Devanagari	95	95	100.00%
Gujarati	148	140	94.59%
Gurumukhi	102	98	96.08%
Kannada	32	32	100.00%
Malayalam	41	41	100.00%
Oriya	48	48	100.00%
Roman	107	105	98.13%
Tamil	34	34	100.00%
Telugu	61	61	100.00%

The consistent performance of the word segmentation results indicates the suitability of the proposed method. The actual word count is manually obtained ground truth from the dataset and the number of words obtained after segmentation is listed in the segmented words column in the above table. The results show perfect segmentation over different scripts including Devanagari, Kannada, Malayalam, Oriya, Tamil, and Telugu. A small number of words were incorrectly segmented in other scripts including Bangla, Gujarati, Gurumukhi and Roman.

This section introduced a practical and adaptable word segmentation methodology. It can be used to reduce dependence on manual annotation. This method achieved good performance for a broad range of handwritten Malayalam documents and across different Indic scripts. Due to traditional image processing techniques, issues arise when dealing with highly degraded input images or complex layouts. A major drawback is the lack of standardized ground truth data, which impedes a comprehensive evaluation and comparison of segmentation quality. To address these issues, future work could aim at collecting benchmark datasets and providing data-driven approaches with higher precision and better generalisation, such as deep-learning-oriented methods.

3.5 SELECTION OF WORD CLASSES, WORD LABELING AND MHWD CONSTRUCTION

The final lexicon used in this study was produced by directly examining the set of segmented word images obtained from the FIS. Automatic clustering approaches were attempted but found unreliable for consistent lexical grouping in this corpus, so class selection was carried out manually on the segmented-word collection. All segmented word images were manually inspected to identify the most frequently occurring words across the dataset. The search focused on selecting words that (i) appears repeatedly in multiple documents, (ii) are contextually relevant to FIS records, and (iii) has a sufficiently large sample size to support supervised learning. Based on these criteria, a total of 321 word classes were finalised for dataset construction.

However, the number of samples present in these 321 word classes was initially imbalanced. Some labelled word classes contain only a few representative word images. To construct a balanced dataset suitable for robust training and evaluation of recognition models, additional samples were collected through a targeted data acquisition effort. Handwritten word samples for the selected 321 classes were gathered from individuals in the age group of 10 to 65 years. It ensures diversity in handwriting styles and representation of diverse demographics. Figure 3.8 illustrates word samples collected from multiple individuals during this phase.

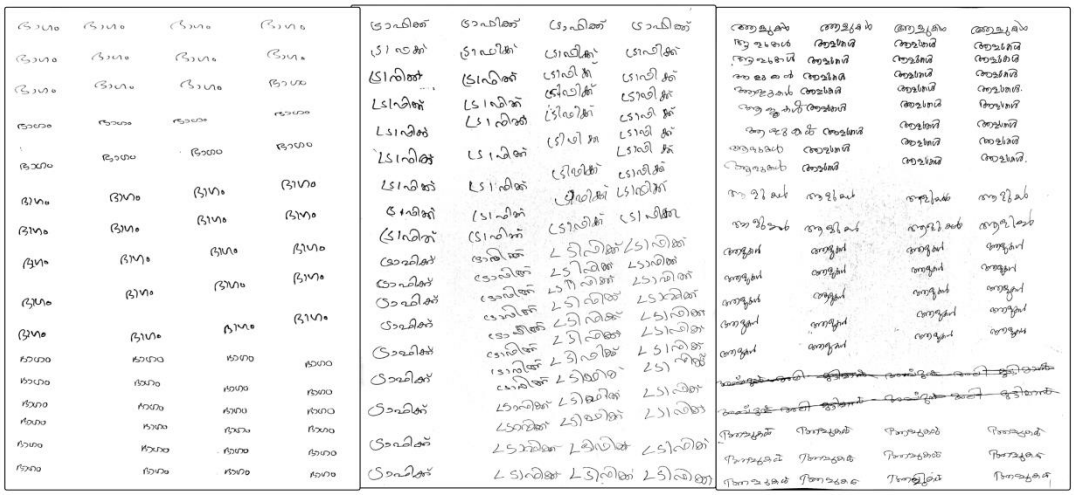

Figure 3.8 Word Samples Collected to make the Dataset Balanced

The newly collected handwritten data is a document containing word samples. On these documents, the same preprocessing and segmentation pipeline that was previously applied to the FIS documents was performed. After segmentation, word images were manually checked and put into the right class of words, thus increasing a sample count in each word's class. This process was iterated until the size of each of the 321 word classes reached 150 samples (full balanced dataset).

The result of this procedure is the MHWD, with 321 word-based handwritten word classes and containing 150 visually verified images per word instance. Such a balanced and curated collection is an important benchmark to train and test handwritten word recognition system in Malayalam script.

3.6 DATASET CHARACTERISTICS AND STATISTICAL OVERVIEW

Statistical profile of the selected handwritten Malayalam word classes in the data set is presented in Table 3.3. Each row of the figure is for a single word class, and is shown both in Malayalam script and its Manglish counterpart.

Table 3.3 Statistics Summary of Image of Word Classes from the Handwritten Dataset (MHWD)

Word_Class_Mal	Word_Class	Max_Height	Min_Height	Avg_Height	SD_Height	Max_Width	Min_Width	Avg_Width	SD_Width	No:of Ascendant	No:of Descendant	Character Count
ആളുകൾ	aalukal	188	60	120.00667	21.0708319	613	177	437.573333	78.199603	1	3	5
ആൾ	aal	157	34	92.446667	20.1793745	313	81	225.213333	46.2765	1	1	2
ആശാരി	aashari	205	76	130.3	32.5168162	557	322	413.88	42.96059	1	1	5
ആരും	aarum	188	69	113.16	23.8866992	432	251	321.126667	39.375847	0	2	4
ആദ്യത്തെ	aadyathe	174	51	116.50667	20.0172415	641	211	508.586667	66.193775	0	2	5
ആൺ	aan	169	73	117.77333	19.6662644	481	257	337.306667	34.302759	1	1	2
ആകെ	aake	159	66	100.02	18.7930732	483	289	370.006667	42.348002	0	1	3
ആരോ	aaro	147	44	99.26	15.8931558	512	232	407.406667	44.281689	0	1	4
ആളെ	aale	146	62	92.053333	17.5163872	506	295	380.633333	49.479884	0	2	3
ആണ്	aanu	198	47	117.36667	27.8736235	467	99	348.393333	53.539256	1	1	3
ആയതിൽ	aayathil	196	47	127.84	27.5569906	704	220	499.5	83.947901	2	1	5
ആവാം	aavaam	153	73	98.94	12.773008	514	307	373.393333	39.810702	0	1	4
ആയതിന്	aayathinu	188	50	131.31333	22.6580778	776	209	545.4	84.562245	2	1	5
ആയതിനാൽ	aayathinaal	224	61	122.65333	22.9279994	844	427	592.693333	75.633145	2	1	7
അതോവത്തിൽ	abavathil	201	57	118.64	25.0048742	918	253	687.3	94.15397	2	0	8
ആയത്	aayathu	158	47	114.18667	19.0743761	529	141	383.98	55.11654	1	1	3
ആത്മഹത്യ	aathmahathya	162	73	106.44	18.2437131	620	243	452.686667	66.855978	0	2	6
ആശുപത്രി	aashupathri	225	95	124.98	22.1306635	831	473	559.933333	72.326359	1	3	6
ആയ	aaya	154	35	94.926667	16.6353426	426	114	307.686667	45.09784	0	1	2
ആയി	aayi	203	97	137.16	19.2083593	390	227	298.993333	31.697213	1	1	3
അമ്മ	amma	136	40	86.786667	18.0907294	447	97	302.086667	66.946042	0	0	3
അമിത	amitha	180	35	113.54	26.1507501	577	129	382.146667	90.434314	1	0	4
അബ്ദുൾ	abdul	207	46	123.48	28.4058022	589	198	452.026667	88.840902	2	1	4
അച്ചന്റെ	achante	212	57	133.95333	22.1910903	845	205	535.693333	88.775518	1	2	6
അദ്ദേഹം	adheham	141	39	106.18	17.5905543	648	141	461.593333	72.800925	0	1	6
അടി	adi	180	54	124.74667	18.571371	369	76	264.44	48.424578	1	0	3
അമ്മയെ	ammaye	125	42	88.28	12.8473707	649	217	448.76	75.393694	0	0	5
അഡ്മിറ്റ്	admit	155	86	116.74667	13.5150221	544	261	406.713333	61.224324	3	1	6
അടിച്ചു	adichchu	162	54	116.32	15.41312	561	142	360.373333	73.721598	2	1	5
അകത്ത്	akaththu	146	52	108.38	16.5217715	602	179	434.4	74.169895	1	0	4

അനുജൻ	anujan	204	43	119.71333	27.710007	705	159	514.246667	112.57074	1	2	4
അംശം	amsham	117	31	77.96	12.4380492	566	139	349.36	75.809083	0	0	4
അനുമതി	anumathi	216	61	135.89333	20.2820599	655	235	522.106667	67.344799	1	1	5
അമ്മയും	ammayum	149	36	102.7	22.542922	622	110	417.266667	101.60975	0	1	5
അന്വേഷിച്ചു	anweshichu	190	83	129.68	20.0443907	802	505	681.36	65.958601	1	3	8
അന്നിയന്റെ	aniyante	174	36	127.69333	21.2652915	678	118	533.673333	89.92823	2	1	7
അങ്ങനെ	angane	131	40	88.56	14.0619961	694	185	538.9	69.103087	0	0	5
അന്നിയത്തി	aniyathi	231	77	140.66	25.3978556	780	297	605.9	67.492098	2	0	7
അമ്മയുടെ	ammayude	175	50	118.78	21.0370689	742	195	589.106667	91.413576	0	1	6
അന്നിയൻ	aniyan	174	36	114.21333	22.0851645	647	118	486.346667	77.355283	2	0	5
അപകടം	apakadam	142	38	88.946667	16.2356754	600	155	482.153333	80.319175	0	0	5
അറിയാം	ariyam	146	52	97.933333	16.5483802	650	172	441.88	78.116658	1	0	6
അറിഞ്ഞു	arinju	189	59	116.98667	32.9555229	680	360	495.206667	72.700875	1	1	5
അപേക്ഷ	apeksha	197	33	109.13333	32.9101133	653	152	415.373333	114.31993	0	0	5
അറിയാൻ	ariyaan	152	48	117.03333	14.3524291	670	238	513.48	50.442802	2	0	6
അപകടത്തെ	apakadathe	156	50	93.3	18.0013888	843	308	679.766667	87.131427	0	0	7
അറിഞ്ഞ	arinja	180	44	114.04667	19.6001145	699	213	491	75.251224	1	0	5
അപ്പോൾ	appol	172	42	108.96	20.7466881	606	203	453.46	65.499377	1	1	5
അപേക്ഷിക്കുന്നു	apekshikkunnu	181	45	131.72667	23.8710136	879	300	744.946667	98.47428	1	2	10
അപകടത്തിൽ	apakadathil	173	53	108.7	21.3746735	891	281	704.773333	117.07873	2	0	8
അസുഖം	asukham	165	66	110.54	18.4732708	610	347	480.006667	63.930222	0	1	4
അറ്റാക്ക്	attack	154	85	115.91333	16.0037232	513	303	409.04	45.934864	1	1	6
അതിനാൽ	athinaal	150	67	105.76	18.3871984	698	303	545.533333	84.960749	2	0	6
അതിൽ	athil	158	55	91.826667	22.0743431	591	280	387.446667	52.335907	2	0	4
അത്	athu	165	50	119.07333	22.029101	417	165	273.853333	51.547827	1	0	2
ഓട്ടോറിക്ഷ	autoriksha	161	63	105.90667	16.1225915	710	453	577.3	52.370061	1	1	8
അതിന്റെ	athinte	186	48	123.55333	26.7096828	613	187	431.82	67.934878	2	1	6
ഓട്ടോയിൽ	autoyil	181	89	121	17.8710194	661	359	500.766667	55.633373	2	1	7
അശ്രദ്ധ	ashradha	128	56	93.48	14.1547259	617	165	444.9	70.440211	0	1	5
അറിയിച്ചു	ariyichu	188	54	131.01333	22.3645513	686	207	501.98	72.450946	2	2	7
ബഹളം	bahalam	135	67	95.466667	13.058671	546	321	425.373333	56.651631	0	1	4
ബാങ്ക്	bank	171	59	106.55333	18.4541004	409	201	277.893333	45.591687	1	0	4
ഭയങ്കര	bayankara	135	71	94.946667	12.9788477	523	307	391.92	48.911828	0	0	5
ബീച്ച്	beach	169	82	111.2	15.7907146	354	192	242.613333	26.474588	2	1	4
ഭാര്യ	bhaarya	159	88	116.98	15.1351555	352	179	254.633333	28.043161	0	1	4
ഭാര്യയും	bhaaryayum	190	91	124.99333	20.2834897	568	256	407.633333	77.43672	0	2	6
ബേപ്പൂർ	beypore	195	105	128.80667	15.6348741	692	338	460.906667	70.473101	1	2	5
ബാങ്കിൽ	bankil	181	84	121.97333	17.6793464	536	320	426.053333	42.178318	2	0	6
ബസാർ	bazar	157	81	110.50667	16.1124162	519	334	405.366667	36.70775	1	0	4
ബന്ധുക്കൾ	bandhukkal	159	69	111.71333	18.536212	604	268	472.44	83.790491	1	0	6
ബന്ധുവായ	bandhuvaaya	170	71	113.45333	17.9586884	682	366	553.433333	66.065464	0	1	6
ഭീഷണി	bheeshani	171	73	108.83333	16.5341734	477	231	325.446667	43.200083	2	0	5
ഭർത്താവ്	bharthaavu	177	93	127.2	16.5150436	651	375	520.02	61.645272	1	0	6
ഭയന്ന്	bhayannu	164	39	102.74	15.2876552	504	133	344.213333	45.603302	1	0	4
ബുദ്ധിമുട്ട്	budhimutt	200	70	150.14667	20.3002091	635	208	485.766667	52.468202	2	3	7
ബോഡി	body	165	77	106.94667	17.3065639	534	256	351.713333	44.389238	1	0	4
ബസ്സിൽ	bussil	145	75	100.78	15.7238121	448	252	322.38	37.471175	2	1	5
ഭവൻ	bhavan	171	84	110.11333	16.6014203	414	271	331.173333	21.861304	1	0	3
ബസ്സ്	bus	233	57	145.88	32.1856945	445	115	289.733333	61.655567	1	1	3
ഭർത്താവിനെ	bharthavine	173	87	127.16	18.4061874	775	417	669.773333	74.589378	1	0	9
ബ്ലോക്ക്	block	190	72	119.96667	18.8748216	529	198	402.433333	48.226468	1	1	5
കാറിന്റെ	carinte	208	103	151.16667	18.6613385	578	331	442.593333	52.393651	2	1	7
ചെന്നതിൽ	chennathil	172	45	129.24667	19.4555688	784	242	604.72	73.170315	2	0	7
ചീത്ത	cheetha	209	86	136.4	30.3657702	480	338	396.306667	27.524037	1	0	4
ചാർജിലുള്ള	chargilulla	210	67	128.41333	28.6710508	908	241	704.42	96.88311	1	2	8
കേസിൽ	casil	200	70	126.28	23.3997493	543	184	409.713333	59.394763	2	0	5
ചവിട്ടി	chavitti	158	80	107.80667	13.7761614	357	182	264.726667	30.239466	2	1	6
കാർ	car	257	55	131.64	32.5617936	357	86	235.446667	56.519205	1	0	3
അവരെ	avare	145	45	90.713333	26.2478791	529	304	402.166667	44.908116	0	0	4
അവന്റെ	avante	214	84	130.44	20.8989569	530	282	408.426667	60.516592	1	1	5
അവകാശം	avakasham	157	50	78.34	14.5418156	679	354	553.573333	56.517649	0	0	6
അവനോട്	avanodu	164	62	109.76	18.6557158	669	338	501.053333	78.240125	1	0	5
അവൻ	avan	195	37	114.16667	30.6447204	463	112	363.32	69.384563	1	0	3
അവധി	avadhi	184	72	117.69333	23.0963047	526	288	423.473333	40.909933	1	0	4
അവളെ	avale	170	55	103.45333	21.109425	591	288	411.886667	79.61604	0	1	4
അവർകൾ	avarkal	151	60	99.606667	16.4255479	593	360	466.093333	46.530327	2	0	5
അവനെ	avane	168	46	103.24	24.1372962	529	360	429.56	37.164585	0	0	4
അവൾ	aval	119	56	78.553333	12.4870795	450	167	361.933333	42.923291	1	0	3
അവിടേക്ക്	avidekku	149	53	99.44	17.0283998	601	335	475.673333	52.749534	2	0	7
അയാളെ	ayaale	117	54	77.34	13.8312834	623	349	461.966667	54.839817	0	1	5
അവിടെയുള്ള	avideyulla	227	81	135.72	29.4966936	902	406	664.426667	101.64345	1	2	8
അയാളുടെ	ayaalude	174	92	124.57333	17.1702637	739	348	510.44	78.023285	0	1	6
അയാൾക്ക്	ayaalkku	153	82	106.34	14.0037281	591	351	491.44	48.965904	2	0	6
അവസരത്തിൽ	avasarathil	155	71	112.17333	18.0387533	844	520	709.86	69.398274	2	0	8
അവർക്ക്	avarkku	141	87	107.53333	10.1072032	573	326	462.766667	57.927819	2	0	5
അയാൾ	ayaal	185	83	118.32	22.4105094	517	356	441.006667	38.988459	1	0	4
അവരുടെ	avarude	182	92	124.94	16.0786525	572	265	413.586667	50.701635	0	1	5
അവിടെ	avide	172	57	95.746667	24.8207404	503	274	374.326667	54.53482	1	0	5
അയൽവാസി	ayalvaasi	186	107	139.88	17.395371	825	426	692.446667	80.833123	2	0	7
അയക്കുന്നു	ayakkunnu	176	78	124.23333	21.5559479	671	438	519.853333	46.087654	0	2	6
ബഹു	bahu	148	76	108.45333	16.0218961	443	264	345.433333	38.667198	0	1	2
ഭാഗം	baagam	137	31	93.48	20.4511516	380	116	293.76	46.254611	0	0	4
ഭക്ഷണം	bakshanam	142	76	101.05333	11.526657	603	307	482.866667	49.741688	0	0	5
ബന്ധപ്പെട്ട	bandhapetta	137	68	96.966667	12.9246105	595	383	498.686667	47.78886	0	2	8
ഭാവിയിൽ	baaviyil	182	83	123.37333	21.9337021	672	326	489.18	68.037448	3	0	7
ബലമായി	balamaayi	151	62	110.37333	18.7220179	589	308	448.853333	56.293799	1	0	6
ഭാഗത്ത്	baagaththu	165	82	112.72	16.1295257	547	298	422.926667	50.541481	1	0	5

കേസ്സ്	casse	241	97	136.94667	27.1607036	433	181	295.026667	43.394001	1	1	4
ബൈപ്പാസ്	bypass	126	51	83.846667	16.6536229	615	326	443.62	63.854174	1	1	6
കാറിൽ	caril	189	38	130.35333	19.9483121	534	176	397.246667	44.951372	2	0	5
ചെറിയ	cheriya	165	57	116.48	21.4073881	535	233	390.293333	48.600898	1	0	5
ചെയ്യുന്നു	cheyunu	179	65	133.52	20.444957	692	217	514.26	70.798346	0	3	6
ചെയ്യാൻ	cheyyaan	134	60	92.4	13.7268593	441	265	331.586667	38.422075	1	1	6
ചേർന്ന	chernna	170	34	98.146667	18.4327197	531	159	399.566667	49.105793	1	0	5
ചെയ്യുന്ന	cheyyunna	166	54	122.28	18.7883368	642	209	487.713333	51.495674	0	2	6
ചെയ്ത്	cheyth	249	97	149.26667	33.5703573	613	225	358.44	77.340156	1	1	4
ചേർന്ന്	chernnu	171	42	113.94667	19.2640552	575	179	414.08	52.80196	2	0	5
ചെവിക്ക്	chevikku	194	67	132.17333	24.6364896	662	205	466.48	63.300734	2	0	6
ചേർക്കുന്നു	cherkkunnu	135	71	97.353333	13.9738502	657	358	483.46	71.457365	1	2	7
ചെക്ക്	cheque	114	64	87.326667	9.60312218	418	186	249.253333	43.467104	1	0	4
ചില	chila	157	70	124.78667	16.739011	370	128	274.3	26.019672	1	0	3
ചോര	chora	113	45	87.133333	9.80181389	394	132	327.286667	29.475037	0	0	3
സിറ്റി	city	261	89	160.00667	32.1754558	413	185	279.18	45.48935	2	1	5
ക്ലാസ്സ്	class	198	99	141.24667	20.7929272	467	253	327.966667	42.356018	1	2	5
കോളേജ്	college	221	55	140.05333	24.4182409	713	273	473.213333	80.453099	1	1	5
ചോദ്യം	chodyam	144	67	98.1	17.2934477	432	250	322.6	30.64376	0	1	5
ചുവടെ	chuvade	205	93	132.65333	22.2851181	565	229	361.213333	65.062492	0	1	4
കോളേജിൽ	collegil	147	78	104.60667	15.8774039	634	326	490.213333	59.838403	2	1	7
ചികിത്സ	chikilsa	122	73	93.606667	10.5892377	548	312	408.62	50.480844	2	0	6
ചുരിദാർ	churidar	179	100	136.93333	17.9913559	771	374	529.193333	106.68556	2	1	6
ധരിച്ച	daricha	178	47	130.36	20.2972182	449	195	340.873333	33.217726	1	1	5
സൈക്കിൾ	cycle	136	71	92.586667	12.8831604	480	274	374.313333	45.058649	2	0	6
ക്രിമിനൽ	criminal	257	93	140.79333	38.1308793	708	420	515.006667	64.501938	3	1	7
കോപ്പികൾ	copykal	169	59	119.36667	18.6552644	707	281	545.66	58.93667	2	1	7
ദേശം	desham	94	54	71.486667	6.86414517	329	210	254.22	23.151636	0	0	4
ക്രൈം	crime	115	52	67.406667	10.5287522	338	175	260.266667	36.035661	0	1	4
ദേഹം	deham	124	36	94.7	11.3188633	469	125	360.113333	37.593712	0	0	4
കമ്മിറ്റി	committee	227	109	148.81333	24.1945963	481	293	385.22	42.185285	2	1	7
കോളനി	colony	169	61	119.76	14.0701481	539	207	358.1	33.264546	1	1	5
കോഴ്സ്	course	190	90	131.67333	23.2541026	608	343	443.34	45.139315	2	1	4
ഡ്രൈവിംഗ്	driving	180	91	127.38	19.3558501	699	436	543.68	44.966998	2	1	7
എടുത്തു	eduthu	199	70	135.54667	24.227694	579	183	462.606667	49.972645	0	2	4
ഡോക്ടറെ	doctore	126	85	101.96667	7.26766048	590	433	508.466667	30.010369	0	1	6
ഡോക്ടർ	doctor	154	81	118.00667	12.1663452	524	310	381.28	39.227392	1	1	5
ഡ്രൈവർ	driver	161	50	96.873333	18.6208831	559	197	388.08	65.700991	1	1	5
ദേശീയ	desheeya	199	103	142.32	22.0690794	531	317	381.786667	37.349982	1	0	5
ഡ്രൈവറുടെ	driverude	146	58	104.43333	14.2662383	713	238	554.96	64.956999	0	2	7
ദൂരം	dooram	161	81	113.73333	14.7239789	372	220	288.533333	33.194712	0	1	3
ഡ്യൂട്ടി	duty	268	101	154.67333	24.3152892	481	282	361.253333	41.654962	1	3	5
ദിവസം	divasam	186	77	116.47333	25.5511765	557	224	384.98	50.269072	1	0	5
ഈ	ee	131	80	100.62	10.2525899	248	168	199.94	16.325534	0	1	1
ഫറോക്ക്	feroke	168	44	108.34	22.9276631	617	180	472.06	74.486708	1	0	5
എനിക്ക്	enikku	197	53	122.79333	29.0792702	577	163	370.126667	86.319739	2	0	5
എന്നയാൾ	ennayaal	191	43	109.50667	31.1342784	846	176	572.526667	156.36209	1	0	6
എലത്തൂർ	elathur	163	34	98.766667	21.2917564	539	159	399.073333	73.967479	1	1	5
ഏകദേശം	ekadesham	158	47	91.72	14.7788678	792	181	572.526667	87.978383	0	0	6
എന്റെ	ente	185	40	118.06	34.5431575	481	129	303.073333	87.114798	1	1	4
എന്നെ	enne	113	33	67.633333	19.2102114	393	113	245.72	73.909144	0	0	4
എന്ന	enna	121	30	82.326667	23.5865772	480	111	288.46	84.137636	0	0	3
എന്ന്	ennu	159	39	100.72667	19.1846802	405	136	261.633333	42.535854	1	0	3
ഹോട്ടൽ	hotel	150	60	88.88	19.0899345	438	251	331.766667	42.114988	1	1	5
ഗോപാലൻ	gopalan	188	52	111.70667	19.8570044	737	225	570.766667	60.841041	1	0	6
ഇന്ന്	innu	165	48	115.52667	28.5383243	437	94	254.713333	66.677966	1	1	3
ഇൻസ്പെക്ടർ	inspector	205	51	135.3	26.6965041	938	238	684.393333	107.21262	3	2	9
ഇവിടെ	ivide	169	52	124.93333	19.4660959	491	146	397.293333	59.996283	1	1	5
ജില്ല	jilla	211	119	152.23333	16.6362723	385	233	310.133333	26.023494	1	2	4
ഹാജരായി	hajaraayi	221	65	144.39333	29.2868336	865	305	571.106667	127.36651	1	1	7
ഇതിന്	ithinu	164	84	107.08667	12.6179431	445	287	353.96	29.18056	2	0	4
ഹാർട്ട്	heart	210	61	136.91333	21.3141382	515	144	397.986667	49.418618	2	1	5
ഇവർ	ivar	163	41	115.78	18.0679717	426	130	318.786667	40.287399	1	1	3
കണ്ണ്	kannu	145	52	117.99333	9.48647927	295	101	250.713333	23.667513	1	1	3
കാരണം	kaaranam	133	41	99.573333	13.7653413	548	177	447.986667	51.074976	0	0	5
കണ്ടു	kanduu	155	82	125.45333	18.7120591	402	258	318.106667	22.062078	0	1	3
കളവ്	kalavu	168	84	118.82	14.1275476	382	232	293.633333	30.299707	1	1	3
കാണിച്ചു	kaanichuu	224	98	149.10667	26.9201651	661	376	486.12	61.115183	1	2	6
കണ്ടാൽ	kandaal	154	89	112.81333	13.7721393	568	381	449.453333	29.882121	1	0	5
ജ്യേഷ്ഠന്റെ	jyeshttante	256	61	147.64667	23.2975354	811	223	575.906667	87.638374	1	4	9
കല്പണിയാണ്	kalpaniyaanu	212	93	146.36667	21.0128268	914	547	705.846667	71.23405	2	1	8
ജോലി	joli	221	42	141.12667	30.4548402	492	141	363.8	49.894021	1	0	4
കൈ	kai	112	37	90.24	10.378619	293	91	233.933333	27.485673	0	0	2
കരച്ചിൽ	karachil	205	85	130.66667	21.181963	530	365	435.346667	34.675445	2	1	6
കഴുത്തിലെ	kazhuthile	186	89	128.22	18.7758249	707	355	544.86	65.640844	1	1	7
കസബ	kasaba	148	76	99.953333	12.7123754	437	274	330.993333	25.147961	0	0	3
കോടതി	kodathi	161	90	122.84	13.2252687	527	319	406.286667	45.144706	1	0	5
കിടക്കുന്നതും	kidakkunnathu	185	81	123.18667	21.5101175	801	468	686.7	59.475009	1	2	9
കഴുത്തിൽ	kazhuthil	207	90	134.02667	23.5284074	642	397	523.24	46.541692	2	1	6
കെട്ടഴിച്ചു	kettazhichu	177	91	132.70667	17.8308896	566	352	462.8	47.013331	1	4	8
കേട്ടു	kettu	181	50	129.57333	21.2067117	342	77	261.96	43.355566	0	1	4
കത്തി	kathi	171	93	122.68667	16.4430884	375	253	294.04	21.993902	1	0	4
കയർ	kayar	142	76	103.47333	14.3975445	373	241	300.433333	25.43211	1	0	3
മടിയിൽ	madiyil	170	106	128.04	10.6231696	552	310	412.54	48.425837	3	0	6
കുറ്റം	kuttam	157	83	118.18667	14.6403036	392	187	288.82	34.436428	0	2	4
കൂടെ	koode	147	98	118.52	8.28470076	279	201	237.153333	15.693199	0	1	3
കൊണ്ടു	kondu	150	78	109.92667	15.4984286	567	320	421.48	38.948123	0	1	4
ലോറി	lori	210	105	152.92	25.3196946	478	277	355.18	43.950437	1	0	4
മാല	maala	125	42	88.986667	11.1301313	338	90	252.6	28.715385	0	0	3
മാറ്റി	maatti	211	104	145.41333	21.746168	343	209	260.32	28.703616	1	1	5

കോവൂർ	kovoor	161	56	131.32667	12.215562	440	160	379.373333	32.038008	1	1	4
കോടതിക്കും	kodathikkum	191	91	138.90667	19.5483492	679	466	558.826667	41.844274	1	1	8
കോഴിക്കോട്	kozhikode	233	55	143.88667	29.2293315	789	214	552.873333	109.44127	2	0	8
മെഡിക്കൽ	medical	249	89	141	26.2906828	789	309	503.693333	87.480279	2	0	7
മോർച്ചറി	morchari	193	104	144.50667	18.5489431	685	377	491.186667	47.065966	2	1	7
മേൽകാര്യം	melkaryam	183	51	118.77333	20.8911933	677	216	550.753333	61.720546	1	1	8
മകൻ	makan	141	65	97.38	17.5566398	370	208	285.78	34.234265	1	0	3
മാവൂർ	mavoor	148	58	123.1	10.9469935	364	200	304.653333	23.682057	1	1	4
മണിക്ക്	manikku	161	41	97.006667	29.2849214	473	127	282.106667	81.784281	2	0	5
മരണപ്പെട്ട	maranappetta	159	81	110.29333	17.4403925	674	449	544.026667	52.893345	0	2	8
മണി	mani	221	53	133.80667	31.2857149	349	115	249.806667	48.016066	1	0	3
മകൾ	makal	120	72	89.813333	7.93589874	299	185	228.56	23.841834	1	0	3
മകളുടെ	makalude	181	93	122.12667	16.705407	624	377	454.08	42.644659	0	2	5
നടക്കാവ്	nadakkavu	177	80	110.22	19.622392	596	357	437.62	54.537286	1	0	6
മോട്ടോർ	motor	150	56	123.24	11.9195526	522	181	443.04	41.627055	1	1	6
മുറിവ്	murivu	164	49	135.56667	17.2411974	370	117	312.28	35.291381	2	1	4
മൃതദേഹം	mruthadeham	176	57	122.62667	14.9610814	615	218	520.1	47.155735	0	1	6
മൊഴി	mozhi	237	49	132.08	33.90841	489	115	289.82	73.324218	1	0	4
മുൻവശം	munvasham	191	100	125.49333	12.0486495	667	343	471.62	90.905935	1	1	5
മുമ്പ്	mumpu	178	89	132.56	19.4930689	357	201	270.753333	24.72433	1	1	3
മുതൽ	muthal	194	88	123.46	23.7497032	406	231	324.213333	33.050787	1	1	3
നടപടി	nadapadi	196	92	127.5	18.8685806	642	337	446.1	85.231978	1	0	5
മോഷണം	moshanam	124	66	90.54	13.5674267	625	407	531.666667	42.044368	0	0	5
നൽകി	nalki	162	87	110.71613	12.889483	387	211	300.425806	28.967156	2	0	4
നിർത്തി	nirthi	158	45	124.59333	17.2669614	493	159	408.446667	49.794047	3	0	6
ഞാൻ	njan	213	38	110.30667	33.3714342	425	105	265.173333	64.910887	1	0	3
നടത്തി	nadathi	168	76	109.6	20.5309522	420	259	327.6	31.128337	1	0	5
നൽകാൻ	nalkaan	167	68	96.233333	15.5329399	497	326	412.846667	36.678556	2	0	5
നഷ്ടപ്പെട്ടു	nashtapettu	153	81	114.00667	15.2352646	620	375	499.153333	43.417468	0	4	8
ഞാനും	njanum	165	51	112.82667	22.828855	512	165	421.166667	39.042441	0	1	4
നോക്കി	nokki	170	54	126.16667	18.2422282	537	152	402.84	65.887994	1	0	5
നീതി	neethi	185	96	134.10667	21.0070295	380	193	264.2	41.195469	2	0	4
നിന്നും	ninnum	205	37	127.58667	30.6461932	557	119	317.84	78.277718	1	1	5
പരാതി	paraathi	190	51	122.67333	28.8498404	539	175	433.486667	57.815884	1	0	5
പാലം	paalam	103	42	75.666667	9.51746932	433	139	365.186667	28.832941	0	0	4
ഓടിച്ച	odicha	192	81	140.83333	28.6476099	431	225	322.253333	42.840197	1	1	5
പണം	panam	131	30	89.7	17.8339564	477	138	385.36	66.108727	0	0	3
നോട്ട്	note	148	59	122.26	11.3404468	349	136	304.773333	23.129533	1	1	4
നമ്പർ	number	197	61	124.44	25.9857602	441	143	310.346667	60.997266	1	0	4
ഓടിച്ചെന്ന്	odichennu	202	88	146.17333	25.2599147	778	501	633.1	58.045241	2	1	8
ഓടി	odi	179	95	131.02	14.8971899	291	202	239.673333	17.162943	1	0	3
ഒരു	oru	213	81	127.58	24.619036	365	153	248.853333	50.922606	0	1	2
ഒളിപ്പിച്ചു	olippichu	199	89	134.29333	18.6109454	649	408	529.713333	50.699946	2	4	8
പിന്നെ	pinne	180	43	109.72	20.5202729	533	193	382.606667	49.209199	1	0	5
പരിക്ക്	parikku	131	79	102.07333	10.7710084	449	277	343.1	35.497746	2	0	5
പോയി	poyi	206	102	148.16	23.2857839	577	309	420.993333	51.836988	1	0	4
പോലീസ്	police	205	32	119.46667	40.0049441	641	152	433.44	138.20808	2	0	5
പറഞ്ഞ	paranja	173	38	102.80667	20.701271	645	167	433.806667	92.798398	0	0	4
പറയാൻ	parayaan	163	62	101.00667	18.3192055	549	338	437.693333	42.444779	1	0	5
പ്ലാസ്റ്റിക്	plastic	207	94	139.075	24.7895013	674	320	482.08125	68.060357	2	2	8
ഫോൺ	phone	166	40	108.95333	19.8968462	476	137	379.473333	48.578966	1	0	3
പരാതിയുണ്ട്	paraathiyundu	214	50	136.1	20.3105063	779	237	651.606667	66.425085	2	1	8
പിടിച്ചു	pidichu	187	78	121.94667	23.9729811	554	270	394.18	48.51475	2	2	6
പ്രാവശ്യം	pravashyam	196	87	116.52667	20.9503211	606	392	498.126667	49.378578	0	2	7
രാജൻ	rajan	188	48	127.95333	20.8023514	479	148	380.406667	46.447547	1	0	4
രജിസ്റ്റർ	register	233	49	141.29333	33.8456391	721	145	430.913333	108.75305	2	2	7
രാത്രി	raathri	182	87	126.14667	21.6802173	395	256	316.493333	28.333431	1	1	5
റോഡിൽ	roadil	158	53	116.3	14.626517	581	195	474.025	49.934451	1	2	5
പുലർച്ചെ	pularche	152	89	110.88667	12.3204636	667	469	558.453333	32.051955	1	0	6
റോഡ്	road	171	81	128.02667	21.004427	526	292	382.153333	41.46223	0	1	3
പ്രകാരം	prakaaram	118	38	81.713333	10.6247112	532	109	389.326667	46.401221	0	1	6
പ്രായം	praayam	111	65	86.98	8.66523321	388	165	325.14	26.183972	0	1	5
പബ്ലിക്	public	173	92	126.38667	16.3261699	600	372	465.606667	45.409529	2	1	5
ശേഷം	shesham	124	35	80.193333	12.9216597	431	116	319.186667	37.372608	0	0	4
രൂപ	roopa	226	35	121.02	21.4766447	362	115	256.113333	36.10587	0	1	2
ശരിയാണ്	shariyaanu	257	53	133.54667	36.0554363	721	165	428.533333	111.36377	2	0	6
സ്റ്റേഷനിൽ	stationil	169	39	91.146667	29.6842016	547	177	365.193333	93.691244	2	1	8
ശരീരം	shareeram	176	45	127.44667	18.0150073	450	126	335.02	37.006571	1	0	5
സംഭവം	sambavam	173	75	95.653333	13.71908	593	356	468.226667	45.216243	0	0	5
സമയം	samayam	177	32	94.926667	33.5726271	581	106	284.566667	96.842926	0	0	4
സമയത്ത്	samayaththu	143	41	91.453333	19.0548635	475	168	338.68	58.361439	1	0	5
സമീപം	sameepam	162	52	123.44667	18.2426375	493	169	387.76	56.787637	1	0	5
സ്റ്റേഷൻ	station	241	42	132.2	37.4425783	721	160	397.32	117.25524	1	1	6
തല	thala	116	47	86.273333	10.6701432	267	102	225.546667	16.728653	0	0	2
താമസം	thaamasam	138	69	93.613333	15.0903	600	359	465.173333	53.949204	0	0	5
സ്ത്രീ	sthree	244	113	180.86667	23.689285	262	148	192.473333	22.536695	2	1	4
സബ്	sub	135	38	105.42	14.3927158	429	97	294.513333	57.164644	1	0	2
സ്വീകരിച്ചു	sweekaricchu	219	121	167.80667	19.9009871	757	448	571.266667	54.961401	2	3	8
സുമാർ	sumaar	208	40	112.14667	42.0065688	457	111	272.946667	87.138571	1	1	4
സ്വയം	swayam	214	34	107.22	17.3827002	418	113	330.573333	39.490648	0	1	4
സ്ഥലം	sthalam	140	63	93.98	12.515308	489	288	374.313333	38.56633	0	0	4
താക്കോൽ	thakkol	188	87	112.74667	16.4418923	542	357	472.566667	41.209451	1	0	6
തടസ്സം	thadasam	164	73	106.43333	17.481959	487	311	390.22	35.996828	0	1	5
തട്ടി	thatti	204	103	158.87333	18.3587569	305	180	237.826667	25.613732	1	1	4
തിയ്യതി	thiyyathi	205	44	116.94	38.0037244	453	128	287.72	83.445481	2	1	6
താലൂക്ക്	thalookku	160	96	133.26	10.2978509	518	361	429.746667	28.886487	1	1	5
തരുവാൻ	tharuvaan	222	47	144.00667	32.2830186	882	205	603.206667	130.39066	1	1	5
ടിയാന്റെ	tiyaante	211	56	126.6	25.4854207	537	193	406.5	55.961743	2	0	7
ട്രാഫിക്	traffic	263	93	132.04667	22.2046051	583	355	458.013333	50.96443	2	1	6
തുടങ്ങി	thudangi	238	98	152.14	23.2608484	594	336	426.826667	53.964031	1	0	5

തരാൻ	tharaan	191	49	122.7	22.5446963	522	144	413.74	51.456704	1	0	4
തമ്മിൽ	thammil	171	40	135.83333	16.8587926	523	108	426.633333	46.730492	2	0	5
തരണം	tharanam	100	61	76.786667	7.10360159	504	299	429.82	31.881253	0	0	4
വന്നിരുന്നു	vannirunnu	230	86	126.70667	24.3328438	734	471	581.92	62.08838	1	2	7
ഉള്ള	ulla	113	33	98.573333	9.56266815	342	156	268.6	26.328692	0	3	3
വരും	varum	164	35	103.84	20.3286596	329	103	253.986667	28.447375	0	1	3
വീട്ടിൽ	veetil	253	46	127.60667	43.2504176	441	106	284.906667	81.300541	3	1	6
ഉടനെ	udane	202	49	106.10667	17.1052221	602	138	425.886667	76.681205	0	1	4
വാഹനം	vaahanam	119	62	85.06	12.4146312	621	384	521.146667	41.992442	0	0	5
വീടിന്റെ	veedinte	193	61	140.20667	21.3804573	564	186	419.773333	53.02429	3	1	7
വായിച്ചു	vaayichu	297	60	148.54	40.2766483	677	145	401.513333	102.96062	1	2	6
വണ്ടി	vandi	155	47	117.42	12.5672962	352	127	299.6	29.597748	1	0	4
വയസ്സ്	vayassu	214	81	152.16667	30.9604945	467	155	361.686667	31.18913	1	1	4
വില്ലേജ്	village	195	112	146.10667	17.9120617	558	354	423.513333	39.012603	1	1	6
വിധം	vidham	128	93	108.36667	6.2486443	314	257	283.14	12.313694	1	0	4
വിവരം	vivaram	164	46	116.21333	15.1228686	501	157	368.073333	46.525849	1	0	5
വേണ്ട	venda	149	48	102.51333	15.1985686	483	128	358.466667	40.192315	0	0	4
വീട്ടിലാണ്	veettilanu	180	108	140.65333	14.4554427	652	392	497.84	51.605953	3	1	8
വിവാഹം	vivaham	184	92	122.56	19.8206895	768	339	488.08	89.556948	1	0	6
വീഴ്ത്തി	veezhthi	195	119	145.38667	12.6421447	536	342	439.64	36.626999	3	0	6
വാർഡിൽ	wardil	193	56	127.73333	19.7648397	739	218	528.4	63.892514	3	0	6
വേണ്ടത്	vendath	161	74	112.86	17.9358226	564	372	444.913333	36.541472	3	0	5
വില	vila	189	56	119.17333	16.7203456	313	110	243.7	23.072278	1	0	3
വെസ്റ്റ്ഹിൽ	west hill	186	108	142.02	18.377693	745	400	518.466667	49.788307	3	1	8

The table contains several geometric and structural attributes which are computed from all of the samples for each class. These properties are maximum, minimum, average and standard deviation of the heights and widths. This information informs about the spatial properties of the handwritten word image. The height value helps in quantification of the vertical aspect of word images and the width values describe their horizontal extent. A high standard deviation of these values, seen in some classes such as aashari (ആശാരി), apakadathil (അപകടത്തിൽ) can attract the attention of individual differences in handwriting behavior.

Further, the table also lists occurrences of ascendent and descendent characters for each Malayalam word class. These types of components are not unusual in Malayalam script and are crucial for the board-based analysis (i.e. the vertical complexity) of a word. Ascendants rise above the upper mean line, and descendants fall beneath the baseline of handwritten word images. Word classes with multiple ascendants or descendants are more likely to have greater vertical variation in height, which impact segmentation and recognition. The last column shows the total number of characters in each word class. This count is of special use in determining the complexity of a word. The longer words have more segmentation boundaries and higher probability of being overlapped or distorted, so that they are not easy to be accurately recognized.

The information in this figure cumulatively demonstrates the diversity and variation in handwritten samples. Such variability highlights the importance of strong

preprocessing and recognition methods that can handle varying writing styles and word lengths.

Figure 3.9 illustrates a swarm plot that visualises how the average width of handwritten Malayalam words varies with the number of characters in each word class. Horizontal axis is the count, number of the character, and vertical axis is the average width (in pixels) of handwritten word images. Every dot corresponds to an individual part of speech.

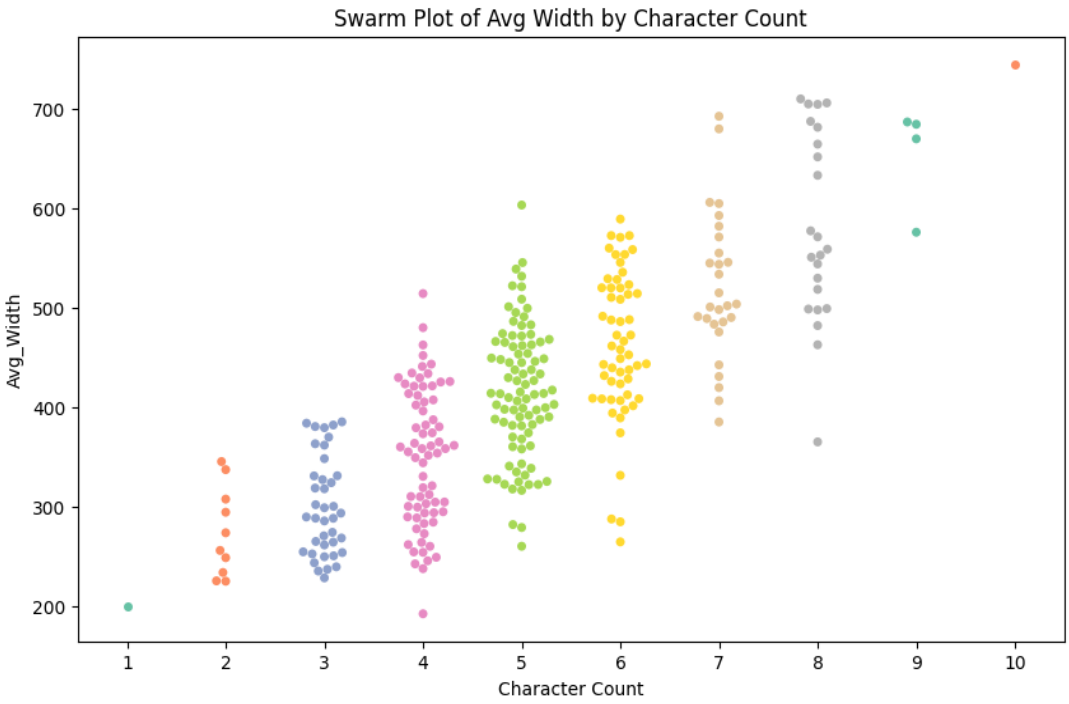

Figure 3.9 Swarm plot showing the distribution of average widths of handwritten Malayalam word images across different character counts

Unsurprisingly, we observe that according to our plot the average width tends to increase with characters, or in other words – longer words are meant to occupy more horizontal space. However, there is substantial spread to the width values at each character count reflecting the variation in writing styles and inter-character spacing and individual character shapes. Consider, for example, at character counts 5 and 6 the width varies between below 300 pixels and over 600 pixels which means that even words of the same number of characters can vary significantly in their horizontal spread.

This observation highlights the need for preprocessing and normalisation operations that can adapt to variable input data given recognition models. Knowledge of the width of distribution over character counts, spread in chapter proportionalities can be useful to select appropriate input dimensions and for designing models that can generalize across word structures.

3.7 SUMMARY

This chapter has covered the step by step development of the MHWD for construction of Malayalam handwritten word recognition systems. The dataset construction includes the collection of real-world FIS documents, causing the samples to represent natural and unconstrained (free) handwriting styles. The documents were preprocessed via a multi-stage preprocessing pipeline in order to improve the quality of the collected digitized FIS. It handles common complications of scanned handwriting documents such as the background noise, varies contrast and skew alignment.

Following preprocessing, the segmentation of handwritten word images from the document image was achieved through a sequence of contour-based methods. Manual inspection and refinement were performed to select the word classes. A balanced representation across classes was maintained by collecting additional samples from a demographically diverse group of writers. The resulting MHWD encompasses a rich variety of word samples spanning multiple age groups and writing styles. A statistical evaluation of the dataset highlighted significant variations in word dimensions and structural features, underscoring the complexity of handwritten Malayalam recognition.

The MHWD is publicly available as an open-access dataset under the CC BY 4.0 International License. The dataset is archived on Zenodo with a permanent DOI: https://doi.org/10.5281/zenodo.20082979. The archive includes the complete dataset along with supporting files such as the README documentation and class mapping information for the handwritten word classes.

4. TrA-MHWR: TRANSFORMER BASED WORD RECOGNITION MODEL WITH ATTENTION ENRICHED FEATURES

4.1 INTRODUCTION AND OVERVIEW OF TrA-MHWR MODEL ARCHITECTURE

Offline Handwritten word recognition in complex scripts like Malayalam is a considerable challenge due to several factors. The intricate structure of Malayalam characters, including loops, curves, and combined characters, as well as the variability in writing style and the large number of characters present in Malayalam, increases the complexity of recognition. Traditional machine learning and early deep learning methods are effective in recognizing simpler and more uniform scripts. However, it is a struggle to generalize across the diverse writing patterns present in scripts such as Malayalam. Thus, there is a demand of more context-aware recognition models that can effectively model the complex spatial and contextual dependencies in handwritten word images.

In this chapter, we present a new architecture called TrA-MHWR (Transformer-based Malayalam Handwritten Word Recognition), which benefits from the latest trends of transformer model in computer vision applications. Transformers are proposed for sequence modelling in Natural Language Processing (NLP). Later, Transformers are adapted for image understanding tasks through models such as the Vision Transformer. The Data-Efficient Image Transformer (DeiT) is a Vision Transformer architecture that views an image as a sequence of non-overlapping patches and self-attention mechanisms is employed to process it. It enables global context modelling and long-range dependency capture, which are particularly beneficial in handling the variability found in handwritten data. Figure 4.1 shows TrA-MHWR model architecture.

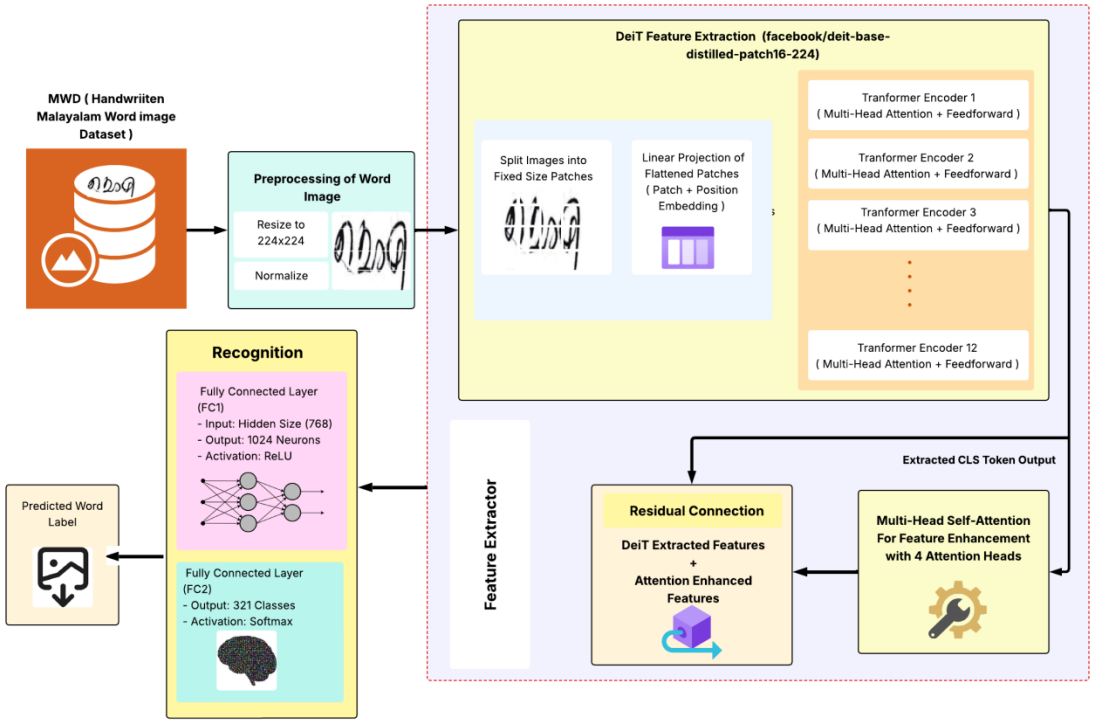

Figure 4.1 Architecture of the proposed TrA-MHWR model. It integrates the DeiT transformer for efficient feature extraction, followed by attention-based feature enhancement and classification using fully connected layers

TrA-MHWR is built upon the DeiT architecture to extract features from the handwritten word image. Due to its data-efficient design, making it highly suitable for tasks where annotated data is limited. DeiT achieves better performance through knowledge distillation and architectural enhancements, thus eliminating the need to rely on a large amount of labelled data. Equipping DeiT within the TrA-MHWR framework enables effective feature extraction from Malayalam word images by converting them into a rich embedding space that preserves both global layout and local features.

For enhancing Malayalam word recognition performance, the TrA-MHWR model utilises an attention layer and a residual connection after feature extraction from DeiT, to perform feature enhancement. This architectural design improves the model's ability to distinguish between closely related word classes, a common challenge in handwritten Malayalam word recognition. The enhanced features are provided to the recognition part, which consists of two fully connected feed-forward neural network layers. One layer takes the embeddings as input features from the feature extractor, and the final

output layer performs classification of 321 word classes using a softmax activation function. Algorithm 4.1 Shows the step-by- step procedure for TrA-MHWR model.

Algorithm 4.1 : TrA-MHWR – Word Recognition Using DeiT + AFFNN

```
Input:
    I: Handwritten word image
    θ: Trained DeiT feature extractor
    φ: Trained AFFNN recognition model

Output:
    y_pred: Predicted class label

I_r ← Resize I to model input dimensions
I_n ← Normalize pixel intensities to [0, 1]

# Feature Extraction
F ← DeiT_Encoder(θ, I_n)

# Multi-Head Self-Attention
A ← MultiHeadAttention(F)
F_att ← F + A          -> Residual connection

# Feedforward Classification
z1 ← DenseLayer1(F_att)
z2 ← DenseLayer2(z1)
y_pred ← Softmax(z2)

return y_pred
```

Thus, the TrA-MHWR model architecture presents a novel approach that combines DeiT feature extraction, attention layer equipped for feature enhancement with a residual connection, and deep learning-based classification layers for final recognition from the enhanced features. This architecture can achieve higher recognition accuracy and model robustness. The following sections detail the individual components of the architecture, experimental setup, and performance evaluation that collectively validate the efficacy of this proposed approach.

4.2 FEATURE EXTRACTION WITH DEIT

In the proposed architecture, initial feature extraction from the handwritten word image is performed using the DeiT, specifically the deit-base-distilled-patch16-224 model obtained from the Hugging Face repository [106]. The DeiT model is an advanced transformer-based backbone pre-trained on the ImageNet-1k dataset, encompassing 1 million images across 1,000 classes, and operates at an input resolution of 224×224

pixels. This model, introduced by Touvron et al.,[107], is a variant of the ViT model [108], specifically optimized for training efficiency and performance on relatively smaller datasets.. DeiT employs both transformer-based modelling and knowledge distillation for superior generalization in low-resource settings.

The DeiT model follows the transformer encoder structure proposed initially by Vaswani et al. [109], adapted for image data. In this model, the input image is represented as shown in Eq. (4.1).

$$I \in R^{H_{img} \times W_{img} \times Ch} \tag{4.1}$$

Here, H_{img} denotes the image height in pixels, W_{img} denotes the image width in pixels and Ch represents the number of color channels. The input image divided into non-overlapping patches of size $P \times P$. For the deit-base-distilled-patch16-224 variant, the input image is of size 224×224, and 16×16 patch size, resulting in $N = \frac{224^2}{16^2} = 196$ patches. Then, each patch is flattened and linearly projected into an embedding vector of dimension D=768, forming the initial patch embedding as shown in Eq. (4.2).

$$X_p \in R^{N \times D} \tag{4.2}$$

Here, N denotes the number of image patches obtained after dividing the input image, and D represents the dimesion of the embedding vector for each patch. Thus, X_p is a matrix containing the patch embeddings, where each of the N rows corresponds to a single patch and each row has a feature dimension of D. A learnable classification token is then introduced as a D-dimesional vector, as shown in Eq. (4.3).

$$x_{cls} \in R^D \tag{4.3}$$

This token is perpended to the patch embedding sequence, resulting in the final input matrix, as shown in Eq. (4.4).

$$X = \left[x_{cls}; X_p\right] \in R^{(N+1) \times D} \tag{4.4}$$

The spatial information is incorporated using a learnable positional encoding matrix $PE \in R^{(N+1) \times D}$, which is added element-wise to the embedding sequence, as shown in Eq. (4.5)

$$Z_0 = X + PE \tag{4.5}$$

The sequence Z_0 is then passed through a stack of $L = 12$ transformer encoder layers. It contains a multi-head self-attention (MHSA) module and a feedforward network (FFN). For any layer l, the output of the MHSA block with residual connection is computed as shown in Eq. (4.6).

$$Z_l' = MHSA\big(LN(Z_{l-1})\big) + Z_{l-1} \tag{4.6}$$

Similarly, the FFN output with residual connection is computed as shown in Eq. (4.7).

$$Z_l = FFN(LN(Z_l')) + Z_l' \tag{4.7}$$

Here, LN denotes Layer Normalisation. The multi-head attention mechanism splits the embedding into $h = 12$ heads, and scaled dot-product attention is performed by each head. The results from these heads are combined and linearly transformed to restore the original dimension.

After processing through the 12 transformer layers, the final output is obtained as $Z_L \in R^{(N+1)\times D}$, which contains the contextualized embeddings for every input token. The embedding corresponding to the classification token, denoted as z_{cls}, as shown in Eq. (4.8).

$$z_{cls} = Z_L[0] \, \epsilon \, R^D \tag{4.8}$$

It is extracted from this output and used for subsequent prediction tasks. The DeiT architecture includes a distillation token x_{dist}, which is used for knowledge transfer during training by mimicking the behaviour of a teacher model. But x_{dist} is not used in this work.

In practice, each word image from the MHWD is resized to 224×224 and normalised to lie in the [0,1] pixel intensity range. The input images were resized to 224×224 pixels to maintain compatibility with the backbone architectures employed in the proposed TrA-MHWR model, particularly the pre-trained convolutional and transformer-based networks. Models such as Vision Transformer (ViT), DeiT, and CNN architectures are commonly designed to accept fixed-size inputs of 224×224, as these models are

originally pre-trained on large-scale image datasets using the same resolution. Adopting this input size enables effective transfer learning by preserving the learned feature representations and network configurations. These preprocessed images are passed through the pretrained deit-base-distilled-patch16-224 model, which outputs the complete sequence of embeddings. This feature vector serves as the input to a downstream attention module followed by dense layers for classification into one of the 321 predefined Malayalam word classes.

4.3 FEATURE ENRICHMENT USING MULTI-HEAD SELF-ATTENTION AND RESIDUAL CONNECTION

After extracting high-level image representations using the DeiT model, further enhancement of these feature representations is performed through a custom-designed feature enhancement module. To further refine and recalibrate the DeiT feature embeddings for script-specific discrimination, an Attention-based Feature Enrichment Module (AFEM) is introduced. This module applies task-adaptive MHSA followed by a residual augmentation step. This design aims to enhance the discrimination capability of the extracted DeiT features, particularly for the handwritten Malayalam word recognition task, where fine-grained differences play a vital role. Through MHSA, the model is able to capture complex intra-token relationships throughout the entire input sequence. It improves the discriminative quality of the feature representation. The residual connection ensures that the original semantic encoding is preserved, enhancing model stability and convergence.

4.3.1 Multi-Head Self-Attention-Based Feature Refinement

Let $I \in R^{B\times3\times224\times224}$be a batch of RGB input word images. The output of the DeiT encoder is a token embedding, which can be represented as shown in Eq. (4.9).

$$Z = Deit(I) = [z_{cls}, z_1, z_2, \dots, z_N] \in R^{B\times(N+1)\times d} \tag{4.9}$$

Here, $z_{cls} \in R^d$ is the classification token that summarizes the global image-level features. $z_i \in R^d$ are the patch tokens derived from the input image. B is the batch size and $d = 768$ is the embedding dimension of hidden size in DeiT-Base [107]. $N = 196$

is the number of non-overlapping image patches for a 16×16 patch size on a 224×224 input.

The AFEM module models global intra-token interactions to capture long-range structural dependencies present in handwriting, such as grapheme overlap, curvature flow, stroke continuation, and script-specific ligatures, which are particularly crucial in Malayalam. For an enhancement of the Deit features, the token matrix Z is passed through the MHSA layer. The MHSA mechanism processes this token embedding by linearly projecting the input sequence into Query, Key and Value (Q, K, and V) matrices. In this Q, K and V can be calculated as shown in Eq. (4.10).

$$Q = ZW^Q, K = ZW^K, V = ZW^V \tag{4.10}$$

$Here, W^Q, W^K, W^V \in R^{d \times d_k}$. Each attention head computes scaled dot-product attention as shown in Eq. (4.11).

$$Attention(Q, K, V) = softmax\left(\frac{QK^T}{\sqrt{d_k}}\right)V \tag{4.11}$$

With h parallel heads, the outputs from all heads are concatenated and passed through a final linear projection, as shown in Eq. (4.12).

$$MHSA(Z) = Concat(head_1, \dots, head_h)W^O, W^O \in R^{h \cdot dk \times d} \tag{4.12}$$

This helps the model to attend to information from multiple representation subspaces, enriching the feature representation by modelling interactions between different token positions [109]. In Malayalam script the model can capture different stylistic patterns such as rounded strokes, straight edges, diacritics, nasal marks (anusvaram), and conjunct characters.

4.3.2 Residual Connection for Stability and Information Preservation

To preserve the core semantic structure learned by the DeiT backbone, a residual connection is incorporated around the MHSA layer. This results in the residual-enhanced representation, as shown in Eq. (4.13).

$$\overline{Z} = Z + MHSA(Z) \in R^{B\times(N+1)\times d} \quad (4.13)$$

Thus, this residual connection ensures that the attention refinement process does not overwrite important learned features obtained from DeiT, but instead augments them. Residual connections have been widely adopted in deep learning due to their ability to stabilise training, facilitate gradient flow, and prevent feature degradation, as demonstrated in ResNet and Transformer architectures [109], [110].

4.4 EXTRACTION OF THE CLASSIFICATION TOKEN AND WORD RECOGNITION VIA FULLY CONNECTED LAYERS

After the feature refinement AFEM module, the final token sequence output is as shown in Eq. (4.14).

$$\overline{Z} = [\bar{Z}_{cls}, \bar{z}_1, \bar{z}_2, \ldots . \bar{z}_N] \in R^{B\times(N+1)\times d} \quad (4.14)$$

It contains the contextualised embeddings for all patch tokens and the classification token. Among these, only the embedding corresponding to the classification token is selected as the representative feature for the entire input word image. This embedding, denoted as $\bar{Z}_{cls} \epsilon\ R^d$, serves as a compact, high-level semantic representation of the input, enriched with global contextual information gathered during both the DeiT encoding and the additional attention refinement. The classification token, initially introduced in Bidirectional Encoder Representations from Transformers (BERT) for classification tasks and adopted by ViT / DeiT, is intended to aggregate information from all patches via self-attention layers [107], [108], [111].

To transform this abstract feature vector f into a class prediction, a lightweight recognition head comprising two fully connected dense layers is equipped. This projection enables dimensionality reduction, nonlinear transformation, and final classification. The process is defined as shown in Eq. (4.15) and (4.16).

$$h = \sigma(W_1 f + b_1) \quad (4.15)$$

$$\bar{y} = softmax(W_2 h + b_2) \quad (4.16)$$

Here, $W_1 \in R^{1024 \times d}$ and $b_1 \in R^{1024}$ are the weight matrix and bias of the first dense layer. σ denotes the Rectified Linear Unit (ReLU) activation function, $W_2 \in R^{C_{out} \times 1024}$ and $b_2 \in R^C$ correspond to the final classification layer. C_{out} is the total number of output classes, set to 321 in this case, representing the number of distinct handwritten Malayalam word classes in the dataset and $\bar{y} \in R^C$ is the resulting vector of class probabilities.

The ReLU activation function provides non-linearity into the transformation process, thus the model learn complex mappings between the features and their corresponding labels. The final softmax layer ensures that the output vector $\bar{y}$ is normalized into a probability distribution over all classes, where the predicted label corresponds to the class with the highest probability.

This two-layer recognition head strikes a tradeoff between the discriminability and computational cost. The intermediate hidden layer (with 1024 units) models a correspondence between the high dimensional feature space and low-dimensional output classes, so it may help to extract some median discriminative features that would be beneficial for robust classification.

In short, the classification token sampled from the enhanced transformer output contributes notably to guide the utterance-level recognition. The constrained use of a non-linear transformation with a subsequent softmax activation function is flexible enough to force the high-dimensional semantic content to be mapped placidly back onto single class label, even in the case that there is high inter-class visual similarity (as typically observed for handwritten scripts).

4.5 EXPERIMENTAL CONFIGURATION AND TRAINING SETUP

All experiments for evaluating the proposed TrA-MHWR model were conducted in a cloud-based computational environment using Google Colaboratory Pro+. It provides access to high-end GPU accelerators. The system was equipped with an NVIDIA A100-SXM4-40GB GPU, known for its exceptional computational capabilities in deep learning and large-scale training tasks. The virtual environment operated on a Linux-based system (Kernel version 6.1.123+), supported by an x86_64 CPU architecture,

83.48 GB of RAM, and 235.68 GB of total storage space, out of which approximately 195 GB was available during runtime. These hardware configurations ensured seamless model execution, particularly suitable for resource-intensive Transformer models.

The experimental setup was implemented using Python version 3.11.13 with a deep learning pipeline constructed using PyTorch. It was also integrated with the Transformers library available from Hugging Face for model handling. In this, OpenCV version 4.12.0 was used for preprocessing the scanned word images, including resizing and normalization steps, while Scikit-learn version 1.6.1 supported model evaluation through performance metric computation. All training routines and evaluations were executed in a Jupyter notebook interface on Google Colab, which offered both GPU acceleration and high RAM capacity, thereby optimizing the training efficiency.

The proposed model was trained using the MHWD, comprising 321 word classes. Word images were resized to 224×224 to match the input resolution expected by the DeiT feature extractor. To ensure balanced evaluation, the MHWD dataset was split into 70% training, 15% validation, and 15% test. The batch size is selected to 32 for training the TrA-MHWR model with a learning rate of 0.0001. Training was performed using the AdamW optimizer, which combines adaptive gradient descent with decoupled weight decay regularisation to enhance convergence. The loss function used was categorical cross-entropy, appropriate for multi-class classification tasks.

Despite the complexity of the model, the training process was computationally manageable due to the effective use of hardware acceleration. The total number of trainable parameters in the complete architecture, including the pretrained DeiT backbone and the attention-enhanced classification head, was 89,869,633. This scale of parameterization reflects the high representational capacity of the model while remaining feasible for execution in a cloud GPU environment.

This experimental configuration is chosen to effectively support the computational requirements of the model, ensuring reproducibility and transparency throughout the experimentation process. By using open-source tools, publicly available cloud

resources, and precise hardware specifications, the study establishes a reproducible and scalable platform for future research in handwritten document analysis and recognition.

4.6 RESULTS AND DISCUSSION

This section explores a detailed analysis of the experimental outcomes achieved using the proposed TrA-MHWR model. The evaluation focuses on assessing the model's effectiveness in recognizing handwritten Malayalam words in MHWD through quantitative performance metrics and qualitative analysis. The section also presents the behaviour of the model across training epochs. Additionally, detailed comparisons and discussions are presented to highlight the improvements introduced by TrA-MHWR. The impact of architectural enhancements, such as attention-based feature enrichment and residual learning, is also discussed in terms of recognition accuracy and generalisation capability.

4.6.1 Metrics for Assessing TrA-MHWR Model Performance

The evaluation of the proposed TrA-MHWR model was conducted using multiple standard performance metrics to ensure a transparent and rigorous assessment of its effectiveness in recognising handwritten Malayalam words across 321 classes. Due to the large number of output classes and the challenges associated with visualising a full confusion matrix, macro and micro-averaged metrics were used to provide an evaluation.

a) Accuracy

Accuracy remains the most intuitive and widely used metric for classification performance. It is the proportion of correctly classified word samples relative to the total number of word samples in the test set. Acurcay can be calculated using Eq. (4.17).

$$Accuracy = \frac{Number\ of\ Correct\ Predictions}{Total\ Number\ of\ Predictions} = \frac{TP + TN}{TP + TN + FP + FN} \tag{4.17}$$

While accuracy offers an overall indication of model performance, it alone may be inadequate in multi-class settings, where a more detailed analysis of per-class predictive behaviour is essential.

b) Categorical Cross-Entropy Loss

Loss functions have an important role in evaluating model performance throughout training and testing. It measures model error, that is, the loss calculates how far the predicted probability distribution is from the actual label distribution. In this work, categorical cross-entropy loss was used, which is the most widely used loss function in multi-class classification tasks. Let $y = [y_1, y_2, \ldots\ldots, y_C,]$ be the one-hot encoded label vector and $\bar{y} = \bar{y}_1, \bar{y}_2, \ldots\ldots, \bar{y}_C$ be the model's predicted probability over C. Then the cross entropy loss can be calculated using Eq. (4.18).

$$L_{CE} = -\sum_{i=1}^{C} y_i \log \bar{y}_i \tag{4.18}$$

The loss function penalises the incorrect prediction of the model heavily when the probability of the wrong prediction is high. In the training process, a sharp reduction in loss indicates effective parameter optimisation, and during validation and testing, low loss values imply better generalisation.

c) Macro and Micro Averaged Metrics

To ensure a better evaluation of the model's recognition capability and effectiveness, both macro and micro averaged values of precision, recall, and F1-score were used in assessing the performance. These metrics are beneficial in multi-class classification tasks. In this work, 321 classes are present and reporting individual class-level results is impractical due to the large number of categories. Instead, these averages provide a concise yet informative summary of the model's overall performance.

Macro Averaging Method

Macro averaging involves computing precision, recall, and F1-score independently for each class and then averaging these values. This method gives equal weight to all classes, regardless of how frequently they appear in the dataset. As a result, macro

metrics reflect the model's ability to perform uniformly across the entire label set. It makes them appropriate for assessing classification fairness and consistency when per-class reporting is difficult. Macro-precision, macro-recall and macro-F1 score is calculated using the Eq. (4.19), Eq. (4.20) and Eq. (4.21) respectively.

$$Macro - Precision = \frac{1}{C}\sum_{i=1}^{C}\frac{TP_i}{TP_i + FP_i} \tag{4.19}$$

$$Macro - Recall = \frac{1}{C}\sum_{i=1}^{C}\frac{TP_i}{TP_i + FN_i} \tag{4.20}$$

$$Macro - F1\ Score = \frac{1}{C}\sum_{i=1}^{C}\frac{2\ \times\ Precision_i \times Recall_i}{Precision_i +\ Recall_i} \tag{4.21}$$

Where C, the total is number of classes and $TP_i, FP_i\ and\ FN_i$ are the true positives, false positives, and false negatives for class i, respectively.

Micro Averaging Method

Micro averaging computes precision, recall, and F1-score by aggregating the total number of correctly and incorrectly predicted instances across all classes. Then the metrics are calculated. Macro averaging treats each class equally, but micro averaging accounts for the relative size of each class. It gives more influence to classes with a larger number of samples. Macro averaging gives a comprehensive measure of overall model performance, making it especially useful when the aim is to assess the classifier's effectiveness over the whole dataset, rather than per class. Micro precision, recall and F1-score can be calculated using Eq. (4.22), (4.23) and (4.24) respectively.

$$Micro - Precision = \frac{\sum_{i=1}^{C} TP_i}{\sum_{i=1}^{C} TP_i +\ \sum_{i=1}^{C} FP_i} \tag{4.22}$$

$$Micro - Recall = \frac{\sum_{i=1}^{C} TP_i}{\sum_{i=1}^{C} TP_i +\ \sum_{i=1}^{C} FN_i} \tag{4.23}$$

$$Micro - F1\ Score = \frac{2\ \times\ Micro - Precision \times\ Micro - Recall}{Micro - Precision +\ Micro - Recall} \tag{4.24}$$

4.6.2 Quantitative Results and Discussion

The proposed TrA-MHWR model was evaluated through a comprehensive training and validation process by running 13 epochs. The training process results show the progressive reduction in training, validation, and test losses, with consistent

improvements in accuracy. Training began with a high loss of 5.6864 and test accuracy being merely 3.46 % in the very first epoch. This small value represents the random prediction of the model in initial learning. During training, the loss values decreased and accordingly increased the classification accuracies. The test accuracy grew rapidly to 98.67% until epoch 5 in Figure. This rise in performance continued to increase and max out at an epoch 13 test accuracy of 99.65%.

The results obtained in epoch-wise validation and test metrics show stable convergence of the model TrA-MHWR. For instance, validation loss decreased from 5.2816 in epoch 1 to 0.0462 in epoch 13, while validation accuracy improved from 3.87% to 99.50%. A similar trend was observed in test performance, where the test loss reduced to 0.0380 and test accuracy reached its peak at 99.65%. This progression shows that the model consistently learned meaningful representations of the data without showing signs of overfitting. The gap between training and validation or test performance is maintained at a minimum across the final epochs, indicating a well-generalised model.

The decision to stop training at 13 epochs was based on empirical evidence. Beyond this point, there is no improvement in the rate of both validation loss and accuracy. It suggested that the model had reached its optimal performance. Early stopping was not applied explicitly, but convergence monitoring guided the epoch selection. Extending the training further showed negligible improvement while increasing the risk of overfitting and computational cost, thereby justifying the chosen stopping point. Figure 4.2 shows the test and validation accuracy of the model.

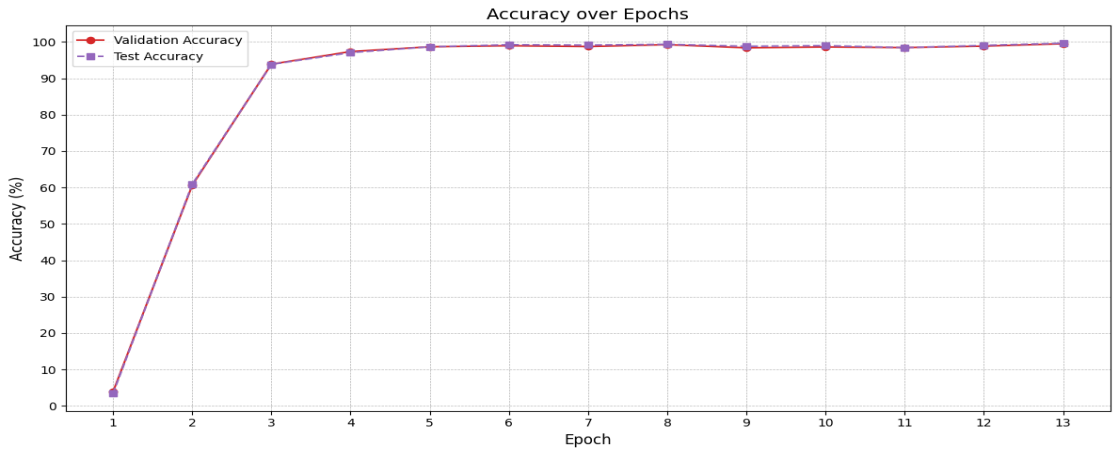

Figure 4.2 Epoch-wise Test and Validation Accuracy of the TrA-MHWR model.

This plot presents the progressive increase in accuracy during model testing and validation in each training epoch. The model reaches high accuracy within a few epochs and shows stable performance towards the final epochs. The figure 4.3 shows the training, testing and validation loss of the proposed TrA-MHWR model.

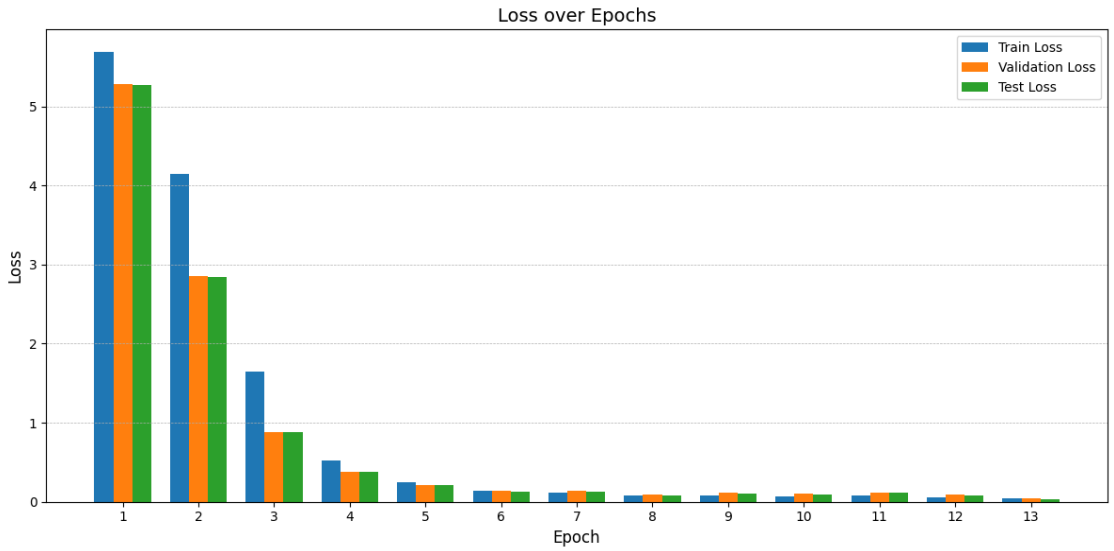

Figure 4.3 Epoch-wise Training, Testing and Validation Loss of the TrA-MHWR model.

To assess classification quality beyond raw accuracy, macro and micro-averaged performance metrics were computed. The micro-precision, recall and F1 were 0.9965 in all metrics, indicating the effectiveness of the model at resolving instances from all of the 321 classes as a whole. These are such scores that even there is a distribution of the class, the processed model predicts well on test-devised set always with high-confidence class. The final performance of the proposed TrA-MHWR model is presented in Table 4.1.

Table 4.1 Final Test Performance Metrics of TrA-MHWR

Metric Type	Precision	Recall	F1-Score
Micro Average	0.9965	0.9965	0.9965
Macro Average	0.9967	0.9963	0.9964

In addition, the macro precision was calculated as 0.9967, macro recall as 0.9963, and macro F1-score as 0.9964. These metrics, which assign equal weight to each class irrespective of its frequency, confirm the uniformity in model performance across all

classes. The minimal difference between macro and micro scores further emphasizes the model's balanced classification capability. This indicates that no particular subset of classes dominated the performance outcome.

Overall, the experimental results validate the robustness and efficiency of the proposed recognition architecture TrA-MHWR. The results show that the integration of DeiT-based patch embeddings, attention-based feature enrichment, and residual learning significantly contributed to the model's discriminative ability. The consistently high scores across all performance indicators underscore the model's capability to distinguish complex handwritten word patterns, making it a strong candidate for scalable handwritten word recognition tasks.

4.6.3 Comparative Evaluation of Model Configurations

In this section, a comprehensive set of experiments was conducted for comparison and to analyze the effectiveness of multiple deep learning architectures in recognizing handwritten Malayalam words. The comparisons focused on evaluating how well each model could learn and represent discriminative visual features from segmented handwritten word images. To ensure fairness in evaluation, different feature extraction networks were paired with selected recognition architectures, forming complete end-to-end recognition pipelines. The main objective of this study was to develop a configuration that delivers high recognition accuracy and exhibits robustness to handwriting variability.

The proposed recognition framework employs the Data-efficient Image Transformer (DeiT) as the feature extraction module. To validate the relevance and suitability of DeiT for handwritten Malayalam word analysis, it was benchmarked against several established Convolutional Neural Network (CNN) architectures and the standard ViT, which serves as a baseline transformer-based design. All comparative studies were performed using the constructed MHWD, ensuring consistency in training, validation, and testing workflows.

The feature extraction networks considered in this evaluation differ substantially in their structural depth, parameter density, feature propagation strategy, and underlying design philosophy. This diversity provides a meaningful basis for understanding how

traditional convolution-based models and transformer-based models behave on handwritten Indic script data. All models were initialized with ImageNet pre-trained weights and adapted for feature extraction by discarding their classification heads. The resulting feature representations were subsequently forwarded to the recognition module without modifying any intermediate architectural components. This enabled a controlled comparison in which performance variations could be attributed primarily to differences in feature extraction capability rather than training conditions or optimization bias.

The feature extractor architectures we choose are DenseNet-121, MobileNet-V2, ResNet-152, VGG-19, ViT and DeiT. These models cover a wide range of deep learning architectures, from computationally cheap designs (e.g., MobileNet-V2) to ultra-deep high-capacity ones (e.g., ResNet-152) and transformer-based self-attention models (ViT and DeiT). For the recognition stage, three architectures were used: a standard Feed-Forward Neural Network (FFNN), a Bidirectional Long Short-Term Memory (BLSTM) network, and an Attention-based Feed-Forward Neural Network (AFFNN). The BLSTM recognizer was selected due to its widespread use in handwriting recognition systems, where sequential dependencies and contextual relationships between visual sub-patterns are relevant. The following subsections are the descriptions of each feature extraction and recognition architecture.

a) DenseNet121

DenseNet-121 (Densely Connected Convolutional Network – 121 layers) was proposed by Huang et al.[112], is a deep convolutional neural network developed to improve feature propagation efficiency, strengthen gradient flow, and reduce parameter redundancy. Conventional CNNs process input only from the previous layer, whereas DenseNet adopts a densely connected architecture in which every layer receives feature maps from all preceding layers. This connectivity structure promotes the reuse of features, which results in rich hierarchical representations expressed by fewer parameters than other deep architectures.

The first layer in the network is a convolution and pooling layer that extracts basic low level features from the input image. After this, the architecture is contains four densely

connected blocks. Each block contains a series of convolutional layers, and within each block, the output of each layer is concatenated with all earlier outputs belonging to the same block. This cumulative feature aggregation ensures that no useful information is lost as the depth increases and that the learning process benefits from both shallow and deep-level features. Between consecutive dense blocks, transition layers are used for dimensionality reduction and downsampling. These transition layers apply batch normalization, convolution, and average pooling to prevent the network from becoming excessively large while maintaining computational efficiency. After the final dense block, the network performs global average pooling to compress feature maps into a feature vector that reflects the learned image characteristics, followed by a fully connected classification layer when used for standard ImageNet classification tasks. Figure 4.4 shows a densnet architecture with 3 dense blocks and transition layers.

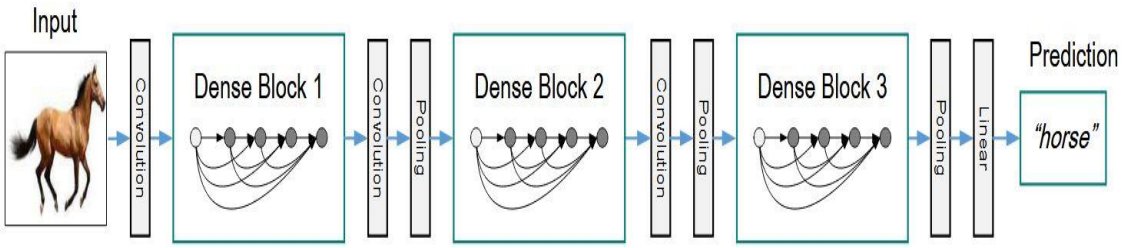

Figure 4.4 Densnet with three dense blocks

When viewed as an architectural diagram, DenseNet-121 can be understood as a sequence of convolutional and pooling operations leading to four dense blocks, separated by transition stages. The dense blocks are increasing in depth were of 6,12,24 and 16 layers with a totality of 121 trainable layers including the classifier. The connectivity pattern of fully convolutional networks is visually represented in the diagram above with highly connected feature map paths projecting from multiple forward directions into any layer within a block. This architecture illustrates the design intuition of reusing and retaining features, instead of relearning common patterns at different depths.

DenseNet-121 is a proper candidate for feature extraction in image-based recognition tasks for its ability to produce very discriminative low-dimensional representations. The high connectivity reduces redundant learning and increases generalization

especially for datasets showing intra-class variations as handwritten data. In addition, the total number of parameters in our architecture is much lower than many other complex deep networks with similar performance, which helps lowers memory requirement and accelerates convergence. Pretrained DenseNet-121 models can be readily obtained, in particular those trained on the large-scale ImageNet dataset; thus, transfer learning is feasible by discarding the final classifier layer and using the global feature vector result as input of any user defined classification unit. As a result, DenseNet- 121 is a promising contender to be compared with other algorithms for feature extraction–based recognition setups such as the handwritten Malayalam word recognition systems.

b) MobileNetV2

MobileNetV2 is a feature contraction network for efficient deployment ideal for environments where computational resources are limited, such as mobile and embedded devices[113]. Instead, it seeks to strike a good balance between computational efficiency, low parameter count and competitive recognition performance. In contrast to the conventional CNN structures, which are predominantly made from standard convolutions, MobileNetV2 introduces architecturally proportional configurations based on depth-wise separable convolutions and an inverted residual structure to decrease the complexity of computational while improving representation capabilities.

The architecture starts with a regular convolutional layer, and contains some number of bottleneck blocks. Each bottleneck block consists of 3 operations: pointwise convolution expanding the channel size, depthwise convolution respectively applying for each channel to acquire spatial information and then pointwise convolution reducing dimensionality of the expanded features. This structure constitutes an inverted residual block, with low input-output dimension but high-dimension expansion at intermediate process. A linear bottleneck is used to prevent representational collapse caused by nonlinear activation of highly compressed feature vectors. Residual (skip) connections are integrated when the input and output dimensions match, allowing adequate gradient flow and improving training stability. Figure 4.5 shows the architecture of MobileNetV2.

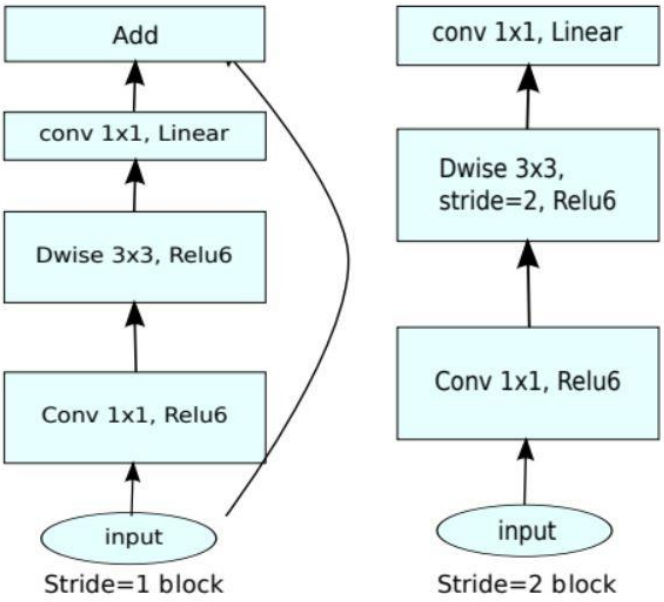

Figure 4.5 MobileNetV2 Architecture

When visualized using its architectural diagram, MobileNetV2 appears as a layered sequence of bottleneck blocks with repeated configurations depending on the required output feature scale. Depthwise separable convolutional modules significantly reduce multiply-accumulate operations compared to classical convolutional operations. The design also includes stride-based downsampling blocks that progressively extract hierarchical spatial features, ultimately leading to a global average pooling layer that condenses feature maps into a compact representation suitable for classification or further downstream processing.

MobileNetV2 is particularly well-suited as a feature extractor for applications requiring low-latency inference and a reduced memory footprint, such as handwritten word recognition, because it delivers high-level discriminative features at minimal computational cost. Its reduced number of parameters and low floating-point operations enable faster training and inference while maintaining robustness against variations present in handwritten data. When applied in a transfer-learning context, pretrained MobileNetV2 weights provide a strong initialization derived from large-scale datasets such as ImageNet, enabling effective adaptation to domain-specific tasks. For these reasons, MobileNetV2 is an appropriate model for comparison with other deep feature

extractors, providing insight into the efficiency-accuracy trade-off in handwritten Malayalam word recognition experiments.

c) ResNet152

The architecture ResNet152 proposed by He et al. [110] ResNet-152 is a very deep convolutional neural network architecture belonging to the Residual Networks family, introduced by He et al., to overcome the degradation and optimization challenges that arise as neural networks grow deeper. Traditional deep CNNs often face vanishing gradients, training instability, and accuracy saturation beyond a certain depth. ResNet-152 addresses these limitations by integrating residual learning, enabling the training of extremely deep models while maintaining strong convergence behaviour and improved generalization performance. With 152 layers, it is one of the deepest ResNet variants and can capture rich hierarchical visual features suitable for complex recognition problems. Figure 4.6 shows the ResNet-152 architecture.

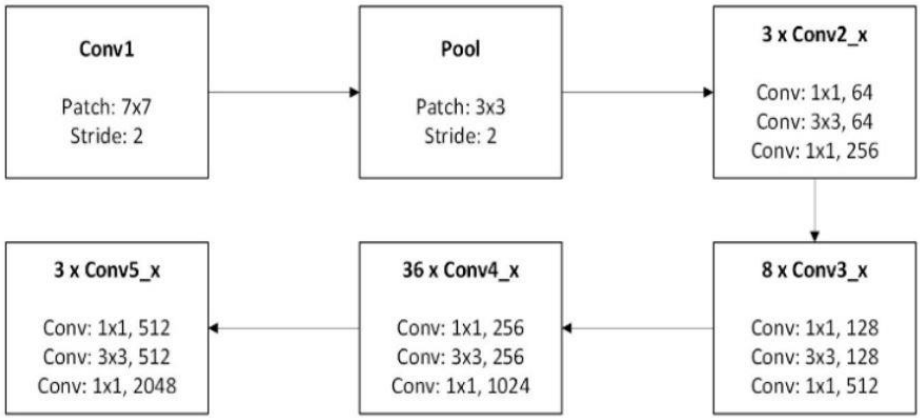

Figure 4.6 ResNet-152 Architecture

The architectural core of ResNet-152 is the residual block, which allows the network to learn the difference between the input and the layer's desired output rather than performing a complete transformation. Each residual block contains convolutional layers and an identity shortcut connection that adds the block's input directly to its output. The skip connection is helpful in that it helps to keep useful information throughout layers and gradients are well propagated backwards through the network during training. As a result, ResNet-152 can learn more discriminative feature representations than conventional networks of similar or smaller size.

Visually, the network architecture is organized as a long sequence of stacked residual blocks built on top of an initial convolutional and max-pooling stage. As such, the number of feature channels is increased and the size (in terms of spatial resolution) of the feature maps decreases as more layers are added. Such hierarchical transformation allows the model to learn increasingly complex and semantically rich representations of input images. The feature maps are global average pooled towards the end of the architecture and fully connected over a dense layer to obtain the final class predictions. Bottleneck residual blocks are being used to make the tremendously deep network computationally feasible, while low dimensional projections are computed before every low and high cost convolutional operations.

ResNet-152 is also commonly used as a backbone feature extractor because its depth allows to learn very discriminative and robust visual features. Pretrained with ImageNet, it has strong transfer learning and generally offers better performance from its downstream tasks. ResNet-152 is suitable for capturing the small stroke-level, shape-level and structural variations that exist in different styles of handwriting in Malayalam script. Its identity mappings can hold the information across multiple layers, which benefits to model complex handwritten patterns and improve recognition accuracy.

d) VGG19

VGG19 is also a well-known relatively simple deep learning model, leading to early adoption in the field of deep learning research [114]. The architecture uses a sequence of small 3×3 convolutional filters, stacked one after another and interleaved with max-pooling layers, to gradually reduce spatial resolution. This repeated use of small kernels, rather than fewer large kernels, increases the network's effective receptive field while preventing the parameter count from becoming excessively large and maintaining a clear, modular structure that is easy to reproduce and analyse. Fugure 4.7 shows the configuration of the VGG network.

ConvNet Configuration					
A	A-LRN	B	C	D	E
11 weight layers	11 weight layers	13 weight layers	16 weight layers	16 weight layers	19 weight layers
input (224 × 224 RGB image)					
conv3-64	conv3-64 **LRN**	conv3-64 **conv3-64**	conv3-64 conv3-64	conv3-64 conv3-64	conv3-64 conv3-64
maxpool					
conv3-128	conv3-128	conv3-128 **conv3-128**	conv3-128 conv3-128	conv3-128 conv3-128	conv3-128 conv3-128
maxpool					
conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256 **conv1-256**	conv3-256 conv3-256 **conv3-256**	conv3-256 conv3-256 conv3-256 **conv3-256**
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 **conv1-512**	conv3-512 conv3-512 **conv3-512**	conv3-512 conv3-512 conv3-512 **conv3-512**
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 **conv1-512**	conv3-512 conv3-512 **conv3-512**	conv3-512 conv3-512 conv3-512 **conv3-512**
maxpool					
FC-4096					
FC-4096					
FC-1000					
soft-max					

Figure 4.7 VGG19 architecture

The network structure consists of multiple convolutional blocks, each containing several convolutional layers followed by a pooling operation. In VGG19, these convolutional blocks are deeper than in smaller VGG variants, resulting in 19 weight layers in total when counting convolutional and fully connected layers. Early layers capture low-level image features such as edges and textures. The deeper layers combine those primitives into higher-level patterns and object parts. The final portion of the network flattens the learned feature maps, passes them through fully connected layers, and outputs class probabilities via a softmax layer when used for classification tasks.

As a feature extractor, VGG19 is often employed by removing the original classifier and using the outputs of a late convolutional block or the penultimate fully connected

layer as a fixed-length image descriptor. These pretrained descriptors, commonly obtained from models trained on large-scale datasets such as ImageNet, provide strong, general-purpose visual features that transfer well to many downstream tasks. For handwritten word recognition, VGG19's earlier convolutional layers reliably capture stroke edges and local patterns. In contrast, mid-level filters capture combinations of strokes and local shapes that are useful for distinguishing similar handwritten tokens.

Despite its representational strengths, VGG19 is relatively heavy in terms of parameters and computational cost compared with more recent architectures. The network's fully connected layers contribute a majority of the parameter counts, inference or fine-tuning needs large memory and computation power. Therefore, in practice, one typically replaces the VGG19 classifier with task-specific layers, performs dimensionality reduction on the learned features, or uses only parts of the network (e.g., solely using a portion while discarding others such as classification trunk) to trade performance for efficiency.

Overall, VGG19 has been a trusty and interpretable architecture for feature extraction. The ease to understand of its simple stacked structure, and the prevalence of pretrained weights (and associated techniques) makes it a natural benchmarking baseline. Exclusively for handwriting-oriented tasks VGG19 accounts for strong low- and mid-level visual features, which can be either fine-tuned or concatenated with lightweight classifiers to reach very competitive recognition performance when computation time is not an issue.

e) Vision Transformer (ViT)

In terms of image representation learning, there has been a huge leap in recent work with the proposition of a fully transformer-based model for visual recognition tasks: the ViT [108]. In contrast with their mainstream CNN counterparts which use spatially local filters to extract the hierarchical features, ViT treats images as sequences of non-overlapping patches and employs selfattention mechanisms to capture global dependencies over the entire image. This paradigm is motivated by the success of transformer models for natural language processing problems, in which attention-based modeling enables capturing long range contextual relationships.

In the ViT architecture, the input image is divided into fixed-size non overlapping patches (for example, 16×16 pixels), and each patch is flattened and linearly projected

into a vector of fixed dimensionality. These patch embeddings can then be stacked into a sequence, just like token embeddings in NLP Transformers. In this sequence, a learnable class token is added to pool all the global image level information. Positional encodings are added to preserve spatial ordering, as transformers do not naturally model location. This spatial shift guarantees that the connections between patches are preserved during the self-attention computations.

The backbone of ViT is a stack of transformer encoder layers that include multi-head self-attention and feed-forward submodules, residual connections and layer normalisation. Self-attention further allows each patch to attend over all patches providing the model with ability to learn relationships beyond local receptive fields which could be potentially hard to express through direct convolutions only. Therefore, ViT constructs powerful representations that efficiently encode global visual context from early processing levels.

The architectural diagram of ViT commonly shows the patch extraction as well as linearly projecting patches, adding the class token, and applying positional embeddings before passing through transformer encoders. This block-wise visualization emphasizes the modular structure of ViT, and demonstrates how individual components participate in hierarchical feature learning without necessarily using convolutional filters. The ultimate class token from the highest encoder layer is utilised for classification; it can be seen as a summary of the whole image.

ViT is well-suited for feature extraction in handwritten word recognition due to its ability to model long-range dependencies and structural variations inherent in handwriting. Additionally, ViT's token-based representation aligns naturally with the need to understand spatial relationships across different regions of a handwritten word, such as ascenders, descenders, or compound characters found in Malayalam script.

In the comparative study of feature extractors, ViT is chosen for its strong performance on fine-grained recognition tasks and its ability to generalise well when used with transfer learning. Its transformer-based design complements CNN-based models, making it a compelling candidate for evaluating feature extraction performance in handwritten Malayalam word recognition.

f) Feedforward Neural Network (FFNN)

The FFNN was equipped as a baseline recognition model to check the effectiveness of sequential and attention-based recognition models. In this the extracted features are passed through two fully connected layers in a unidirectional flow. That is the information is propagated forward without recurrence. The first dense layer performed the dimensionality reduction of the feature vector. If the input feature vector is denoted as x, this operation can be written as a linear transformation followed by a non-linear activation represented as, $h_1 = \sigma\,(W_1 x + b_1)$, where W_1 and b_1 are the trainable weight matrix and bias term. σ () represents the activation function. Here, the non-linearity is provided by the Rectified Linear Unit (ReLU), defined as $\sigma(z) = max(0, z)$. This helps the network to capture more complex decision boundaries.

The output from this hidden layer is then mapped to the target classes by a second fully connected layer represented as, $\bar{y} = W_2 h_1 + b_2$. Here, $\bar{y}$ is the vector of predicted class scores, with each component corresponding to a word class. During training, the model parameters (W_1, b_1, W_2, b_2) are updated using backpropagation to minimise the cross-entropy loss between predictions and true labels. In handwritten word recognition, FFNNs provide a computationally lightweight and stable framework for learning discriminative decision boundaries from extracted image features.

g) Bidirectional Long Short-Term Memory (BLSTM)

Recognising handwritten words often requires capturing sequential dependencies across character structures because the spatial order of strokes and shapes sometimes leads to disambiguating word forms. To address this, Bidirectional Long Short-Term Memory (BLSTM) networks were investigated as a recognition model. Unlike conventional RNNs that process a sequences only in the forward direction, BLSTM architectures operate in both forward and backwards directions [115]. This allows the model to utilize both past and future context while processing the sequence of extracted features, thereby capturing richer structural dependencies.

Let $x = \{x_1, x_2, \ldots, x_T\}$ denote the sequence of feature vectors obtained from the DeiT feature extractor. At each time step t, the BLSTM updates its hidden state as $h_t = BLSTM\,(x_t, h_{t-1})$, where h_t represents the hidden state at time t, which is influenced by both the current input x_t and the hidden state from the previous step. The network processes the sequence in two directions, producing forward hidden state $\overrightarrow{h_t}$ and

backward hidden state $\overleftarrow{h_t}$. These are concatenated to form the final representation at each time step as $H_t = [\overrightarrow{h_t}; \overleftarrow{h_t}]$.

The concatenated hidden states are aggregated, and then the representation corresponding to the final time step is passed through two fully connected layers for classification. The first layer reduces the dimensionality and applies a ReLU activation to apply non-linearity, while the second layer maps the features to 321 class probabilities using a softmax function. Due to this hierarchical structure, the BLSTM can capture both sequential and structural patterns in handwritten words, rendering it especially appropriate for representing writing style features of a particular language. Experimental results on the different model settings are represented in Table 4.2.

Table 4.2 Comparison with Different Model Configurations

Feature Extractor	Recognition Model	Test Accuracy (%)	Micro Precision (%)	Micro Recall (%)	Micro F1-Score (%)	Macro Precision (%)	Macro Recall (%)	Macro F1-Score (%)
DenseNet121	BLSTM	98.87	98.87	98.87	98.87	98.88	98.90	98.85
DenseNet121	FFNN	97.92	97.92	97.92	97.92	98.10	97.85	97.80
DenseNet121	AFFNN	98.96	98.96	98.96	98.96	99.05	98.95	98.94
MobileNetV2	BLSTM	98.65	98.65	98.65	98.65	98.78	98.68	98.61
MobileNetV2	FFNN	97.48	97.48	97.48	97.48	97.61	97.40	97.35
MobileNetV2	AFFNN	98.34	98.34	98.34	98.34	98.42	98.37	98.32
ResNet152	BLSTM	98.69	98.69	98.69	98.69	98.71	98.67	98.62
ResNet152	FFNN	97.07	97.07	97.07	97.07	97.66	97.03	96.77
ResNet152	AFFNN	97.30	97.30	97.30	97.30	97.64	97.21	97.11
VGG19	BLSTM	96.61	96.61	96.61	96.61	96.70	96.62	96.45
VGG19	FFNN	94.85	94.85	94.85	94.85	95.12	94.70	94.60
VGG19	AFFNN	92.65	92.65	92.65	92.65	93.36	92.71	92.61
ViT	BLSTM	96.44	96.44	96.44	96.44	96.69	96.43	96.40
ViT	FFNN	97.12	97.12	97.12	97.12	97.30	97.05	96.98
ViT	AFFNN	98.85	98.85	98.85	98.85	98.92	98.82	98.80
DeiT	BLSTM	97.09	97.09	97.09	97.09	97.26	97.20	97.11
DeiT	FFNN	97.89	97.89	97.89	97.89	97.92	97.80	97.76
Proposed DeiT (TrA-MHWR)	**AFFNN**	**99.65**	**99.65**	**99.65**	**99.65**	**99.67**	**99.63**	**99.64**

Across most feature extractors, BLSTM provides stronger performance than the baseline FFNN, indicating that sequence modelling offers advantages when dealing with spatially distributed handwritten word embeddings. The AFFNN model therefore performs competitively with or better than other methods on most problems by also adding attention-based refinement to static FFNN processing. It is worth mentioning that DeiT and AFFNN has the best overall accuracy of 99.65%, which indicates the

power of transformer based representations in conjunction with attention-based recognition.

In general, the table does show how the interaction between feature extraction and classification architecture is critical for recognition performance. It further indicates that the designed AFFNN does not always better outperform all the baselines, yet offers significant gains when combined with larger transformer features, particularly in our DeiT based model. This study verifies that a simultaneous feature and classifier generation process is vital for good handwritten Malayalam word recognition.

4.6.4 Benchmarking TrA-MHWR Model using Five-Fold Cross-Validation on the CMATERdb2.1.2 Dataset

To evaluate the robustness and generalisation capability of the proposed TrA-MHWR model beyond the primary dataset MHWD, an additional experiment was conducted using the publicly available CMATERdb2.1.2 handwritten bangla word dataset [49]. Direct comparison with existing Malayalam handwritten word recognition systems was limited due to the lack of a publicly available standard benchmark dataset for Malayalam handwritten words. The CMATERdb2.1.2 dataset comprises handwritten Bangla word images of 120 city names from the Indian state of West Bengal, and each class contains 150 samples. The data was collected offline and includes significant variations in handwriting styles influenced by things such as age, educational qualification and occupation. These inherent diversities introduce considerable intra-class and inter-class variation, making CMATERdb2.1.2 a rigorous benchmark for handwritten word recognition systems.

The CMATERdb2.1.2 dataset was selected for cross-script benchmarking due to several structural and experimental similarities with the proposed MHWD. Malayalam and Bangla are both Indic abugida scripts derived from the Brahmic writing system and share several common characteristics, including complex character morphology, rounded stroke structures, vowel modifiers, compound characters, and considerable variations in individual handwriting styles. These similarities introduce comparable challenges in handwritten word recognition, particularly in feature extraction and discrimination between visually similar patterns. Therefore, evaluating the proposed

model on Bangla handwritten words provides an effective measure of its generalization capability across structurally related Indic scripts.

In addition, CMATERdb2.1.2 is a widely used benchmark dataset for research on handwritten word recognition. The dataset structure also closely aligns with the proposed MHWD dataset, as both datasets contain 150 samples per class, thereby ensuring consistency in experimental settings and evaluation procedures. Owing to these linguistic, structural, and experimental similarities, CMATERdb2.1.2 was considered an appropriate benchmark for validating the cross-script adaptability of the proposed TrA-MHWR model.

In this study, five-fold cross-validation experiments were conducted to ensure statistically reliable evaluation. In five-fold cross-validation the dataset is partitioned into five equal subsets or folds. In each iteration, one fold is taken as the validation set and the remaining four folds are used for training. This process is repeated five times, such that every subset serves once as the validation set. The performance metrics are averaged after completing all five iterations. This technique is helpful to mitigate the variability that arises from a specific train-test split. It ensures that the models' performance evaluation is not entirely dependent on a particular partition. This benchmarking analysis aimed to assess the model's consistency and adaptability in another Indic script, Bangla, under identical training and evaluation protocols.

While the primary focus of this experiment was to assess performance on the CMATERdb2.1.2 dataset, five-fold cross-validation was also performed on the Malayalam Word Dataset (MHWD)**.** This was done to provide a direct comparative evaluation of the model's behaviour across datasets of different scripts, languages, and writing styles. Identical model configurations, hyperparameters, and early stopping criteria were maintained across both datasets to ensure consistency and fairness in comparison.

Table 4.2 contains the fold-wise validation accuracy, validation loss, and the number of training epochs for both datasets. The results demonstrate that the TrA-MHWR model achieved consistently high performance on both datasets. On the Malayalam Word Dataset**,** the average validation accuracy reached 98.23% with a standard deviation of

0.73**,** and an average validation loss of 0.0328**.** On the CMATERdb2.1.2 Bangla dataset, the model achieved an average validation accuracy of 99.07% and a validation loss of 0.0355, with a similar standard deviation of 0.72 in accuracy. The slightly higher average accuracy on the CMATERdb2.1.2 dataset may be attributed to differences in dataset complexity or character distribution.

Table 4.3 Five-Fold Cross-Validation Results on Malayalam Word Dataset and CMATERdb2.1.2 Bangla Dataset using TrA-MHWR Model.

Fold	Malayalam Word Dataset (MHWD)			CMATERdb2.1.2 Bangla Dataset		
	Validation Accuracy (%)	Validation Loss	Epochs	Validation Accuracy (%)	Validation Loss	Epochs
Fold 1	99.11	0.0341	9	98.45	0.0494	3
Fold 2	99.14	0.0387	4	99.75	0.0091	6
Fold 3	99.89	0.0052	5	99.67	0.0139	3
Fold 4	99.94	0.0054	2	99.97	0.0005	10
Fold 5	99.96	0.0016	3	99.86	0.0042	5
Average	98.23	0.0328		99.07	0.0355	
Standard Deviation	0.73	0.0287		0.72	0.0264	

The observed performance across folds indicates a high level of model stability and generalisation capability across two distinct scripts, Malayalam and Bangla, without requiring major adjustments to the model architecture. This cross-lingual evaluation substantiates the effectiveness of the proposed approach in handling the challenges inherent in handwritten word recognition.

4.6.5 Benchmarking With Existing Malayalam and Indic Handwritten Word Recognition Models

Previous works in the literature were selected and reproduced to provide meaningful performance references for offline Malayalam handwritten word recognition using MHWD. These two frameworks are relevant to Indic scripts and their methodological significance in handwritten word recognition research.

The first benchmark model is the combination of CNN and SVM of Jino et al. for the handwritten word recognition system for Malayalam [71]. This model uses the features extracted using CNN, followed by an SVM Machine classifier for final prediction. This

represents one of the early supervised approaches designed explicitly for Malayalam word-level recognition. When experimented on the MHWD, this model achieved a recognition accuracy of 87.85%. It indicates its limited capability in handling the complex stroke-level variations and diverse writer styles present in real-world handwritten word samples.

The second benchmark model is HWordNet, introduced by Das et al.[87] for handwritten Bangla word recognition. HWordNet is a CNN-based handwritten word recognition model that learns spatial features directly from word images. The architecture is composed of convolution and pooling layers, which are succeeded by a fully connected feed forward network for classification. The feature representation is mapped to the fixed size vocabulary and a word label is computed by the model. Even though it was designed for a script other than Malayalam, Bangla and Malayalam share many scriptural context cues like compound characters and rounded strokes, visually dense words etc., so the model is a fitting comparison choice. When tested by MHWD, HWordNet obtained an accuracy of 96.69% and presented better feature-learning capability than CNN with SVM. The performance comparison with our model on the MHWD dataset is shown in Table 4.4.

Table 4.4 Comparative Performance of TrA-MHWR with Existing CNN–SVM and HWordNet Models

Model/Method	Dataset Used	Feature Extractor	Classifier	Accuracy (%)
CNN + SVM (Malayalam)	MHWD	CNN Features	SVM	87.85
HWordNet (Bangla)	MHWD	CNN Features	FFNN	96.69
TrA-MHWR (Proposed)	**MHWD**	**DeiT Features**	**AFFNN**	**99.65**

These benchmarking experiments serve two purposes: They establish reference performance levels from recognized literature for Malayalam and Indic scripts. They highlight the need for more advanced representation learning techniques that can address script complexity, writer variability, and morphological richness.

The superior performance of the proposed deep transformer-enhanced architecture further confirms that modern attention-driven, end-to-end learnable systems are more

suited for large-vocabulary handwritten Malayalam word recognition compared to earlier hybrid and purely CNN-driven baselines. Table 4.5 presents a comparative analysis of existing handwritten word recognition methods with respect to dataset characteristics, data augmentation strategies, feature extraction techniques, recognition models, and achieved recognition accuracy. The comparison indicates the increasing adoption of deep learning and transformer-based architectures for handwritten word recognition across different scripts. The proposed TrA-MHWR model achieved superior recognition performance on the MHWD dataset compared to the existing approaches.

Table 4.5 Comparison of state-of-the-art offline HWR studies and proposed work

Source	Dataset	Data augmentation	Feature extraction	Recognition	Accuracy (%)
Bhowmik *et al.* [32]	18,000 samples (Bangla)	Nil	Shape-based Feature Descriptor	SVM	83.64
Roy et al. [37]	16,128 samples (Devnagiri)	Nil	PHOG	HMM	94.51
Malakar *et al.* [88]	18,000 samples (Bangla)	Nil	CNN	CNN model, lottery Ticket hypothesis	91.30
Das *et al.* [87]	18,000 samples (Bangla)	Nil	CNN	FFNN (HWordNet)	96.17
Krishnan et al.[81]	IIIT-HWS (English)	Synthetic rendering	CNN	Holistic CNN (HWNet)	91.58
Stuner et al.[82]	RIMES (French)	Nil	CNN	Cascaded BLSTM	90.63
Almodfer et al. [85]	IFN/ENIT (Arabic)	Nil	MCDNN	MCDNN	91.50
Amrouch et al. [86]	IFN/ENIT (Arabic)	Nil	CNN	CNN-HMM	88.95
Wu et al. [56]	Esposalles, RIMES (Spanish, French)	Rotation, Shearing, Zooming	PE-ResNet	BLSTM	91.97
Jino et al. [71]	29,516 samples (Malayalam)	Rotation, Gaussian noise, Shifting	CNN	SVM	96.90
TrA-MHWR (Proposed)	**MHWD (48,150 samples)**	**Nil**	**DeiT**	**AFFNN**	**99.65**

4.7 SUMMARY

This chapter presented TrA-MHWR, a transformer-based handwritten word recognition model designed to address the challenges of feature representation in Indic scripts, particularly Malayalam. The architecture combined DeiT for feature extraction with an attention-enriched feedforward mechanism and residual connections, enabling effective capture of both global and fine-grained contextual dependencies in handwritten word images. A detailed explanation of the multi-head self-attention mechanism and the role of residual connections in stabilising training was provided. Furthermore, the classification token extraction and recognition pipeline through fully connected layers was discussed.

The models settings, training configurations, different evaluation metrics and how the experiments were set up were explained in depth and followed by a thorough evaluation of performance. The experimental results indicated the importance of TrA-MHWR model. It reached an accuracy of 99.65%, which is higher than the traditional CNN-based methods and transformer baselines. Comparisons with various feature extraction and recognition models demonstrated the effectiveness of TrA-MHWR under different settings. Benchmarking on the CMATERdb2. 1. 2 dataset served as an additional proof of generalization ability of the model. Comparisons with the state-of-the-art in Bengali and Malayalam handwritten word recognition also demonstrated the novelty and effectiveness of our approach.

The results obtained in this chapter validate that the application of transformer-based feature extraction and attention-enriched recognition enhances the performance of handwritten word recognition for Malayalam.

Future research can explore extending the TrA-MHWR framework towards a document-level recognition system by including lexicon-free approaches. Further, lightweight versions of the model to recognize handwritten words may be developed to enable deployment of the model in resource-constrained devices such as mobile devices and other edge devices. Another promising direction lies in integrating cross-lingual transfer learning, using models trained on larger Indic script datasets to boost performance in underrepresented scripts.

To conclude, this chapter introduced TrA-MHWR as a state of the art approach in recognition of handwritten word in Malayalam and open up for further applications in Indic script document analysis and retrieval.

5. DOCUMENT RETRIEVAL SYSTEM WITH LOW-RESOURCE-FRIENDLY MODEL SELECTION

5.1 INTRODUCTION AND SYSTEM OVERVIEW

The Process of handwritten document retrieval is a challenging problem, particularly in the case of the Malayalam language, where variations in handwriting style, stroke order, character spacing, and noise present in the scanned document image create significant complexities. Traditional keyword-based search methods cannot be directly applied to handwritten document images. They require text in a machine-readable format. To overcome this limitation, the proposed system combines deep learning–based handwritten word recognition with a retrieval framework that enables efficient searching of scanned documents.

The document retrieval framework presented in this chapter is designed as a practical extension of the handwritten word recognition system developed earlier in this work. While the proposed word recognition model, TrA-MHWR, described in the previous chapter, achieved state-of-the-art recognition accuracy, it demands considerable computational resources. So it makes it less suitable for resource-constrained platforms. The proposed lightweight model is a hybrid model that utilizes EfficientNet-B0 for feature extraction and an FFNN for classification. This architecture significantly reduces the computational overhead while maintaining high recognition accuracy. So this makes the model more suitable for practical use in low resource devices. In addition, explainability was also incorporated into the recognition process using Gradient-weighted Class Activation Mapping (Grad-CAM), allowing the system to provide visual justification for its predictions.

The recognized words from each document collectively form a structured textual representation that serves as input for document retrieval. To retrieve relevant FIS documents for a given Malayalam query, the framework utilizes a two-stage retrieval strategy. In the first stage, Latent Dirichlet Allocation (LDA) identifies latent topics within the recognized word corpus and assigns a dominant topic to each

document and the input query. This process enables topic-level document filtering and reduces the search space. In the second stage, Term Frequency–Inverse Document Frequency (TF-IDF) weighting assigns word importance scores to the filtered documents, and cosine similarity computes similarity scores between the query and document vectors. Documents are ranked based on these similarity scores, and the most relevant FIS documents are retrieved. By integrating handwritten word recognition with topic-guided document retrieval, the proposed framework offers an efficient, scalable, and practical solution for automated retrieval of handwritten FIS documents.Figure 5.1 shows the overall system architecture of the document retrieval system. Algorithm 5.1 shows the step-by-step procedure for the document retrieval system.

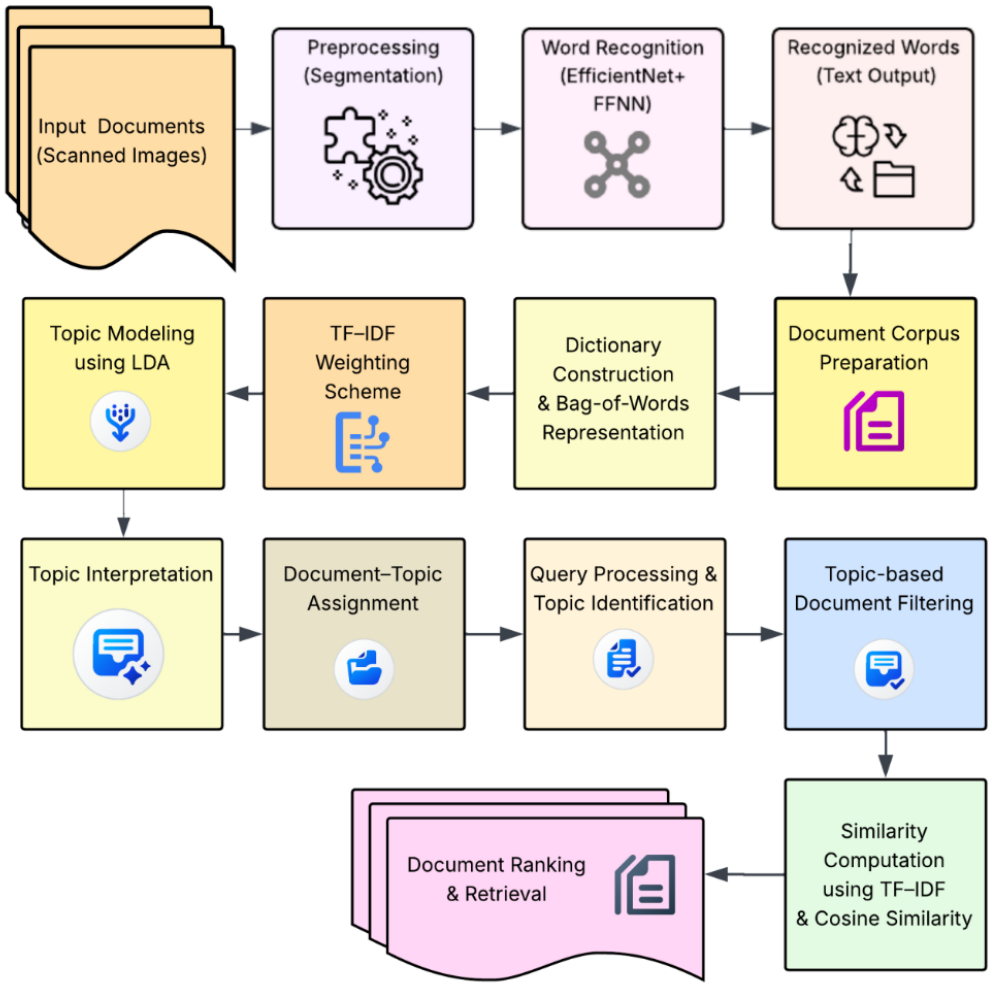

Figure 5.1 Document Retrieval System Architecture

Algorithm 5.1: Document Retrieval Using Recognized Words and Topic Modeling

```
Input:
    D: Set of handwritten document images
    q: User query sentence in Malayalam
    ψ: EfficientNet-based word recognizer
    T: TF-IDF vectorizer
    M: LDA topic model

Output:
    R: Top-k most relevant documents

  Initialize DocumentWordMap ← ∅

  # Preprocessing and Word Extraction
  for each document d in D do
      d_p ← Denoising and binarizing d
      W ← Word level segmentations of d_p
      W_rec ← ∅
      for each word image w in W do
          l ← EfficientNet_Predict(ψ, w)
         Add label l to W_rec
      end for
     DocumentWordMap[d] ← W_rec
 end for

 # Convert back to text list for recognized words
 C ← Construct a corpus for the WordDocumentMap

 # TF-IDF Representation
 V ← T(C)

 # Topic Modeling Using LDA
 Topics ← M(V)

 # Process User Query
 q_p ← Preprocess query q (tokenize, normalize)
 q_vec ← T(q_p)
 q_topic ← M(q_vec)

 # Document Ranking
 for each document d in D do
     score[d] ← CosineSimilarity(q_topic, Topics[d])
 end for

 R ← Top-k documents based on score
 return R
```

5.2 PREPROCESSING AND WORD SEGMENTATION

For the proposed document retrieval system, the handwritten Malayalam FIS must first be transformed into clean and structured word-level image samples suitable for recognition. This is achieved through a preprocessing and segmentation pipeline, which standardizes document images, removes noise, and isolates individual words. These steps ensure that the recognition models operate on consistent and well-prepared inputs. The preprocessing and segmentation steps used in this chapter follow the same methodology described in Chapter 3. That earlier section detailed the procedures for noise removal, contour-based extraction of text lines, and bounding box–based segmentation of words.

Accurate preprocessing and segmentation are essential to the document retrieval application developed in this work. Since the retrieval process depends on recognizing keywords within FIS documents, any misalignment or noise at the word-level input stage would propagate errors into the recognition and retrieval pipeline. Thus, robust preprocessing and segmentation form the foundation of the proposed system, enabling effective downstream word recognition and query-based retrieval.

5.3 EFFICIENTNET-BASED WORD RECOGNITION FOR EDGE DEPLOYMENT

Handwritten word recognition systems in real-world scenarios are frequently implemented on hardware limited platforms such as edge devices or embedded processors. The transformer-based TrA-MHWR model we proposed had the best performances, but its architecture with self-attention mechanism on image patches is quite computationally expensive. To alleviate this issue, an alternative recognition model based on EfficientNet-B0 architecture was created which is a type of convolutional neural network designed specifically to achieve better accuracy with less computation.

5.3.1 EfficientNet-B0 based Feature Extraction

EfficientNet, proposed by Tan and Le [116], uses a compound scaling method that equally scales up the network's depth, width, and input resolution. This design

ensures optimal utilisation of model capacity while maintaining computational efficiency. The base variant, EfficientNet-B0, employs depthwise separable convolutions, squeeze-and-excitation optimisation, and swish activation functions to achieve a strong balance between accuracy and parameter efficiency. When compared to traditional convolutional backbones such as ResNet or VGG, EfficientNet achieves significantly higher accuracy on ImageNet with an order of magnitude fewer parameters and FLOPs [116] . These characteristics make it a suitable model for handwritten word recognition in resource-limited environments.

In this study, EfficientNet-B0 was equipped as the feature extraction backbone of the proposed model. It was initialised with ImageNet-pretrained weights to exploit transfer learning benefits and accelerate convergence. The final classification head of EfficientNet was replaced with a fully connected layer sized to the number of classes in the MHWD, which consisted of 321 word classes with 150 handwritten samples per class. Figure 5.2 shows the block diagram of the recognition pipeline where EfficientNet-B0 is used as a feature extractor and a fully connected layer for handwritten Malayalam word classification.

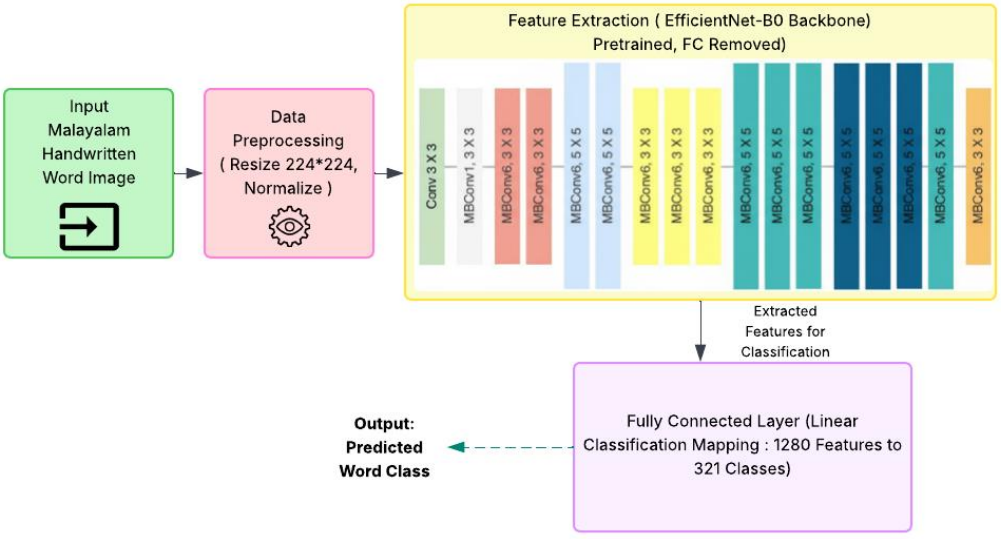

Figure 5.2 Recognition pipeline using EfficientNet-B0 for feature extraction and a fully connected layer for classifying handwritten Malayalam words

The recognition system used in this study integrates EfficientNet-B0 with an FFNN serving as the classifier. This architecture was chosen to achieve a balance between

high recognition accuracy and computational efficiency to make it suitable for resource-constrained environments. The architecture is built from Mobile Inverted Bottleneck Convolutional Blocks (MBConv) with squeeze-and-excitation optimization. As the image propagates through successive layers, the network learns increasingly abstract features from low-level stroke details such as edges and curves to high-level word-specific structures.

At the final stage of convolutional processing, EfficientNet-B0 produces a feature map with 1280 channels. The Global Average Pooling (GAP) layer summarizes each channel by averaging its spatial values and thus creates a compact representation. It reshapes the feature map as a 1280-dimentional feature vecotr as $h(x) = \{ v_1, v_2, v_3, \dots \dots \dots v_{1280} \}$ where $h(x)$ is the embedding of input image x, and each v_i corresponds to the average activation of the i^{th} channel. Such compressed representation will retain key information, but with small dimensions, thereby creating a compact and discriminative embedding pattern for efficient classification.

5.3.2 Recognition with FFNN

The extracted 1280-dimensional feature vector h(x) is fed into a single fully connected FFNN layer mapped directly to the 321 output classes. Let x be the input image, and $h(x)$ is its embedding generated by the EfficientNet feature extractor. The classification layer consists of a linear transformation: $f(x) = W.h(x) + b,$ where W and b give the weight and bias of the classification head. The class prediction is finally computed using the softmax activation function that outputs the probability for each of the 321 classes.

Each word image was resized to 224 × 224 pixels and normalised to the range [0,1], needed matching the input resolution of the EfficientNet. The labels were represented as integers and the dataset was split into training (80%) and testing (20%) sets with distribution maintenance. Cross-Entropy Loss was employed in training the model since multiclass classification problem. We used AdamW as the optimizer because it has a stable convergence and adaptive weight decay. The training was performed for nine epochs with a batch size of eight. For the purpose of convergence control, overfitting protection and debugging, training/validation loss curves were used.

Classification Report:

	precision	recall	f1-score	support
aadyathe	1.00	1.00	1.00	32
aake	1.00	0.86	0.93	36
aal	0.97	1.00	0.99	33
aale	1.00	1.00	1.00	23
aalukal	1.00	0.90	0.95	20
aan	1.00	0.90	0.95	30
aanu	1.00	0.86	0.93	37
aaro	1.00	0.81	0.90	32
aarum	1.00	1.00	1.00	30
aashari	1.00	0.93	0.96	29
aashupathri	1.00	0.97	0.98	29
aathmahathya	1.00	1.00	1.00	26
aavaam	1.00	0.91	0.95	33
aaya	1.00	1.00	1.00	32
aayathil	1.00	1.00	1.00	30
aayathinaal	1.00	1.00	1.00	27
aayathinu	1.00	1.00	1.00	31
aayathu	1.00	0.97	0.99	35
aayi	1.00	1.00	1.00	35
abavathil	1.00	1.00	1.00	29
abdul	1.00	0.96	0.98	25
achante	1.00	1.00	1.00	31
adheham	0.97	1.00	0.98	31
adi	0.88	0.97	0.92	30
adich	1.00	1.00	1.00	37
admit	1.00	1.00	1.00	33
akath	0.91	1.00	0.95	20
amitha	0.79	1.00	0.88	37
amma	0.90	1.00	0.95	35
ammaye	0.97	1.00	0.98	30
ammayude	0.83	1.00	0.91	34
ammayum	1.00	0.96	0.98	26
amsham	1.00	1.00	1.00	23
angane	0.96	1.00	0.98	24
aniyan	1.00	1.00	1.00	34
aniyante	1.00	1.00	1.00	33
aniyathi	1.00	0.97	0.98	32
bussil	0.92	1.00	0.96	24
bypass	1.00	1.00	1.00	32
car	0.91	1.00	0.95	29
caril	1.00	0.96	0.98	27
carinte	1.00	1.00	1.00	22
case	1.00	1.00	1.00	34
casil	1.00	1.00	1.00	34
chargilulla	1.00	1.00	1.00	37
chavitti	1.00	1.00	1.00	20
cheetha	1.00	1.00	1.00	34
chennathil	1.00	1.00	1.00	28
cheque	1.00	0.92	0.96	25
cheriya	1.00	1.00	1.00	31
cherkkunnu	1.00	1.00	1.00	42
chernna	1.00	1.00	1.00	35
chernnu	1.00	1.00	1.00	33
chevikku	1.00	0.86	0.92	21
cheyth	0.88	1.00	0.94	29
cheyunu	1.00	0.97	0.99	38
cheyyaan	1.00	1.00	1.00	34
cheyyunna	1.00	1.00	1.00	33
chikilsa	1.00	1.00	1.00	36
chila	1.00	1.00	1.00	28
chodyam	1.00	1.00	1.00	29
chora	1.00	1.00	1.00	40
churidar	1.00	1.00	1.00	27
chuvade	1.00	1.00	1.00	30
city	1.00	1.00	1.00	28
class	1.00	0.97	0.98	29
college	1.00	1.00	1.00	29
collegil	1.00	1.00	1.00	21
colony	1.00	1.00	1.00	33
committee	1.00	1.00	1.00	28
copykal	1.00	1.00	1.00	35
course	1.00	1.00	1.00	29
medical	1.00	1.00	1.00	39
melkaryam	1.00	0.96	0.98	25
morchari	1.00	1.00	1.00	29
moshanam	1.00	1.00	1.00	30
motor	1.00	1.00	1.00	24
mozhi	0.96	1.00	0.98	26
mruthadeham	1.00	1.00	1.00	34
mumpu	1.00	0.96	0.98	27
munvasham	1.00	0.94	0.97	36
murivu	1.00	1.00	1.00	29
muthal	1.00	1.00	1.00	26
nadakkavu	1.00	1.00	1.00	26
nadapadi	1.00	1.00	1.00	37
nadathi	0.97	1.00	0.99	33
nalkaan	1.00	1.00	1.00	28
nalki	1.00	1.00	1.00	29
nashtapettu	1.00	1.00	1.00	21
neethi	1.00	1.00	1.00	33
ninnum	1.00	1.00	1.00	27
nirthi	1.00	0.97	0.98	32
njan	0.91	1.00	0.96	32
njanum	1.00	1.00	1.00	28
nokki	1.00	1.00	1.00	31
note	1.00	1.00	1.00	30
number	1.00	1.00	1.00	37
odi	1.00	0.81	0.89	31
odicha	0.88	1.00	0.94	22
odichennu	1.00	1.00	1.00	35
olippichu	1.00	1.00	1.00	40
oru	0.81	1.00	0.90	30
paalam	1.00	1.00	1.00	31
panam	1.00	1.00	1.00	29
paraathi	1.00	1.00	1.00	25
paraathiyundu	1.00	1.00	1.00	28
paranja	1.00	1.00	1.00	31
parayaan	1.00	1.00	1.00	29
parikku	0.97	1.00	0.98	31
phone	1.00	1.00	1.00	37
anujan	0.97	0.97	0.97	32
anumathi	1.00	1.00	1.00	27
anweshichu	1.00	1.00	1.00	34
apakadam	1.00	1.00	1.00	28
apakadathe	1.00	1.00	1.00	34
apakadathil	1.00	1.00	1.00	34
apeksha	0.97	1.00	0.98	28
apekshikkunnu	1.00	1.00	1.00	36
appol	0.94	1.00	0.97	30
arinja	1.00	1.00	1.00	36
arinju	0.98	1.00	0.99	41
ariyaan	1.00	1.00	1.00	31
ariyam	0.97	1.00	0.98	32
ariyichu	1.00	1.00	1.00	33
ashradha	0.83	1.00	0.90	19
asugam	0.97	0.97	0.97	30
athil	1.00	1.00	1.00	23
athinaal	1.00	1.00	1.00	31
athinte	0.97	0.97	0.97	33
athu	1.00	0.96	0.98	28
attack	1.00	0.97	0.98	31
autoriksha	1.00	1.00	1.00	28
autoyil	0.93	1.00	0.97	28
avadhi	1.00	1.00	1.00	23
avakasham	0.97	1.00	0.98	32
aval	1.00	0.93	0.97	30
avale	1.00	1.00	1.00	26
avan	0.97	1.00	0.99	33
avane	0.96	1.00	0.98	26
avanodu	1.00	1.00	1.00	24
avante	1.00	1.00	1.00	34
avare	1.00	0.78	0.88	32
avarkal	1.00	1.00	1.00	29
avarkku	1.00	1.00	1.00	31
avarude	1.00	0.97	0.99	35
avasarathil	0.96	1.00	0.98	22
crime	0.85	1.00	0.92	2[illegible]
criminal	1.00	1.00	1.00	2[illegible]
cycle	1.00	1.00	1.00	2[illegible]
daricha	0.98	1.00	0.99	4[illegible]
deham	1.00	1.00	1.00	1[illegible]
desham	1.00	1.00	1.00	2[illegible]
desheeya	1.00	1.00	1.00	2[illegible]
divasam	1.00	1.00	1.00	2[illegible]
doctor	1.00	1.00	1.00	3[illegible]
doctore	1.00	1.00	1.00	3[illegible]
dooram	1.00	1.00	1.00	2[illegible]
driver	1.00	1.00	1.00	2[illegible]
driverude	1.00	1.00	1.00	3[illegible]
driving	1.00	1.00	1.00	2[illegible]
duty	1.00	1.00	1.00	3[illegible]
eduthu	1.00	1.00	1.00	2[illegible]
ee	1.00	1.00	1.00	2[illegible]
ekadesham	1.00	1.00	1.00	3[illegible]
elathur	1.00	1.00	1.00	3[illegible]
enikku	1.00	0.94	0.97	3[illegible]
enna	1.00	1.00	1.00	3[illegible]
ennayaal	1.00	1.00	1.00	3[illegible]
enne	0.93	1.00	0.96	2[illegible]
ennu	0.81	1.00	0.90	2[illegible]
ente	0.97	1.00	0.98	3[illegible]
feroke	1.00	1.00	1.00	2[illegible]
gopalan	1.00	1.00	1.00	2[illegible]
hajaraayi	1.00	1.00	1.00	2[illegible]
heart	1.00	1.00	1.00	3[illegible]
hotel	0.96	1.00	0.98	2[illegible]
innu	1.00	1.00	1.00	3[illegible]
inspector	1.00	1.00	1.00	3[illegible]
ithinu	1.00	1.00	1.00	2[illegible]
ivar	1.00	1.00	1.00	3[illegible]
ivide	1.00	1.00	1.00	2[illegible]
jilla	1.00	0.68	0.81	3[illegible]
joli	1.00	0.93	0.97	30
pidichu	1.00	1.00	1.00	36
pinne	1.00	1.00	1.00	32
plastic	1.00	1.00	1.00	20
police	1.00	1.00	1.00	36
poyi	1.00	1.00	1.00	25
praayam	1.00	1.00	1.00	38
prakaaram	1.00	1.00	1.00	28
pravashyam	1.00	0.97	0.98	32
public	1.00	0.97	0.98	33
pularche	1.00	1.00	1.00	21
raathri	1.00	1.00	1.00	36
rajan	1.00	0.93	0.96	28
register	0.89	1.00	0.94	34
road	1.00	1.00	1.00	24
roadil	1.00	1.00	1.00	25
roopa	1.00	1.00	1.00	33
samayam	0.89	1.00	0.94	33
samayath	1.00	1.00	1.00	37
sambavam	1.00	1.00	1.00	27
sameepam	1.00	0.96	0.98	27
shareeram	1.00	1.00	1.00	36
sheriyaanu	1.00	1.00	1.00	23
shesham	1.00	1.00	1.00	30
station	1.00	0.96	0.98	24
stationil	0.96	1.00	0.98	22
sthalam	1.00	0.97	0.99	40
sthree	1.00	0.96	0.98	27
sub	1.00	1.00	1.00	31
sumaar	1.00	1.00	1.00	28
swayam	1.00	1.00	1.00	34
sweekaricchu	1.00	1.00	1.00	28
thaamasam	1.00	1.00	1.00	17
thadasam	1.00	1.00	1.00	29
thakkol	1.00	1.00	1.00	30
thala	1.00	1.00	1.00	29
thalook	1.00	1.00	1.00	34
thammil	1.00	1.00	1.00	30
avide	0.95	1.00	0.98	21
avidekku	1.00	1.00	1.00	32
avideyulla	1.00	1.00	1.00	32
ayaal	1.00	0.93	0.97	30
ayaale	1.00	1.00	1.00	24
ayaalkku	1.00	1.00	1.00	25
ayaalude	1.00	0.83	0.91	36
ayakkunnu	1.00	1.00	1.00	28
ayalvaasi	1.00	1.00	1.00	31
baagam	1.00	1.00	1.00	33
baagath	1.00	1.00	1.00	28
baaviyil	1.00	1.00	1.00	33
bahalam	1.00	1.00	1.00	29
bahu	1.00	1.00	1.00	28
bakshanam	0.94	1.00	0.97	31
balamaayi	1.00	1.00	1.00	29
bandhapetta	1.00	1.00	1.00	21
bandhukkal	1.00	1.00	1.00	30
bandhuvaaya	1.00	1.00	1.00	24
bank	1.00	1.00	1.00	32
bankil	1.00	1.00	1.00	30
bayankara	1.00	1.00	1.00	33
bazar	1.00	0.90	0.95	30
beach	0.86	1.00	0.93	25
beypore	1.00	1.00	1.00	19
bhaarya	1.00	1.00	1.00	38
bhaaryayum	1.00	1.00	1.00	30
bharthaavu	1.00	1.00	1.00	28
bharthavine	1.00	1.00	1.00	27
bhavan	1.00	1.00	1.00	26
bhayannu	1.00	1.00	1.00	20
bheeshani	1.00	1.00	1.00	28
block	1.00	1.00	1.00	27
body	1.00	1.00	1.00	28
budhimutt	1.00	0.97	0.98	33
_bus	1.00	1.00	1.00	34
jyeshttante	1.00	1.00	1.00	29
kaanichuu	1.00	1.00	1.00	24
kaaranam	1.00	1.00	1.00	26
kai	1.00	0.92	0.96	25
kalavu	1.00	1.00	1.00	29
kalpaniyaanu	1.00	1.00	1.00	36
kandaal	1.00	1.00	1.00	30
kanduu	1.00	0.97	0.98	29
kannu	1.00	0.97	0.98	32
karachil	1.00	0.97	0.98	29
kasaba	1.00	1.00	1.00	27
kathi	1.00	1.00	1.00	15
kayar	1.00	0.97	0.99	34
kazhuthil	1.00	1.00	1.00	28
kazhuthile	1.00	0.96	0.98	26
kettazhichu	1.00	1.00	1.00	29
kettu	1.00	0.97	0.99	35
kidakkunnathum	1.00	0.97	0.99	40
kodathi	1.00	1.00	1.00	24
kodathikkum	1.00	1.00	1.00	38
konduu	1.00	1.00	1.00	44
koode	1.00	1.00	1.00	38
kovoor	1.00	1.00	1.00	40
kozhikode	1.00	1.00	1.00	29
kuttam	1.00	1.00	1.00	32
lori	1.00	1.00	1.00	25
maala	1.00	0.97	0.98	31
maatti	0.92	1.00	0.96	33
madiyil	1.00	0.97	0.99	34
makal	0.97	1.00	0.99	36
makalude	1.00	0.97	0.98	31
makan	1.00	1.00	1.00	27
mani	0.97	1.00	0.98	28
manikku	0.63	1.00	0.78	19
maranappetta	1.00	0.97	0.99	36
mavoor	1.00	1.00	1.00	37
tharaan	1.00	1.00	1.00	30
tharanam	1.00	1.00	1.00	32
tharuvaan	0.97	1.00	0.99	37
thatti	1.00	0.89	0.94	36
thiyyathi	0.97	1.00	0.99	34
thudangi	1.00	1.00	1.00	34
tiyaante	1.00	1.00	1.00	32
traffic	1.00	1.00	1.00	31
udane	1.00	1.00	1.00	29
ulla	1.00	1.00	1.00	31
vaahanam	1.00	1.00	1.00	30
vaayichu	1.00	0.96	0.98	28
vandi	1.00	0.96	0.98	28
vannirunnu	1.00	1.00	1.00	36
varum	1.00	1.00	1.00	21
vayassu	1.00	0.93	0.97	30
veedinte	1.00	1.00	1.00	38
veetil	0.91	1.00	0.95	30
veettilanu	1.00	1.00	1.00	35
veezhthi	1.00	1.00	1.00	29
venda	1.00	0.94	0.97	32
vendath	1.00	0.95	0.97	37
vidham	1.00	1.00	1.00	24
vila	1.00	1.00	1.00	26
village	1.00	1.00	1.00	43
vivaham	1.00	1.00	1.00	28
vivaram	1.00	1.00	1.00	21
wardil	1.00	1.00	1.00	26
west hill	1.00	1.00	1.00	31
accuracy			0.99	9635
macro avg	0.99	0.99	0.99	9635
weighted avg	0.99	0.99	0.99	9635

Figure 5.3 Classification report of the EfficientNet-B0 based Recognition Model

The performance of the EfficientNet-B0 and FFNN recognition model was evaluated using standard classification metrics, and the results are summarized in Table 5.1. The model demonstrated strong overall performance, achieving high accuracy, precision, recall, and F1-score, indicating reliable and consistent recognition across all word classes.

Table 5.1 Performance Metrics of the EfficientNet-B0 + FFNN Recognition Model

Metric	Value
Accuracy	0.9868
Precision	0.9886
Recall	0.9868
F1-Score	0.9869

The model obtained 98.86% precision, suggesting that the model maintained a very low rate of false-positives across word types. The 98.68% recall indicates that the system correctly predicted almost all true examples. Finally, the F1-score was 98.69% indicating an excellent balance of precision and recall.

The high recall and F1-score demonstrate that the model behaves uniformly for the sample of word categories on which it was trained, not over-fitting specific word classes. The findings confirm the generalization capability and applicability of the EfficientNet-B0 with FFNN pipeline in real-world handwritten document recognition and retrieval applications. Table 5.3 displays the results of classification report of the model build.

5.4 EXPLAINABILITY OF THE PROPOSED MODEL

One of the significant challenges in using deep learning models for practical use is their black-box nature. That is the reasoning behind predictions, which is hidden within layers of nonlinear transformations, and it lacks interpretability. In the case of handwritten word recognition, even though the proposed EfficientNet-B0 combined with a FFNN demonstrates consistently high accuracy, the process by which the model distinguishes one handwritten word from another is not directly observable.

Using explainability techniques, the proposed systems become more transparent. It helps to identify which regions or features of the input image have the most significant impact on the model's final decision-making. Such explanations serve several purposes. One advantage is that these methods help researchers and practitioners verify whether the model is focusing on meaningful and relevant handwriting patterns. Explainability is used as a method to detect errors and biases in training. Explainability can be used as a tool for error analysis as well. It can reveal whether misclassifications occur due to limitations in the feature extraction stage or noise in the input image. By applying explainability mechanisms, the recognition system is evaluated not only on its accuracy but also on its ability to provide interpretable insights into its predictions. This alignment with the principles of explainable artificial intelligence (XAI) ensures that the model is suitable not only for research benchmarks but also for deployment in real-world environments where interpretability is as crucial as performance.

Grad-CAM was employed to enhance the interpretability of the model and visualize its rationale behind predictions. It is introduced by Selvaraju et al. [117]. It is now one of the most popular methods to produce visual explanations in convolutional neural networks. Grad-CAM produces a class-discriminative localization heat map to intuitively highlight the important regions in input images for predictions. In the context of handwritten word recognition, this makes it possible to ascertain which distinct stroke, curve or character component the model is looking at when determining that a word belongs to a particular class. The information process during Grad-CAM is illustrated in Figure 5.4.

The fundamental principle of Grad-CAM is based on utilizing the gradient signals that propagate back to the last convolutional layer in the network architecture. While dense layers aggregate information in a way that discards the spatial layout of features, convolutional layers preserve this spatial structure, which makes them highly suitable for identifying the regions in an input image that contribute to the classification outcome. Grad-CAM uses this property by measuring how the class score of interest changes with respect to each feature map in the last convolutional

stage. These gradients are then averaged over the spatial dimensions, producing a set of weights that quantify the relative contribution of each feature map. After that, a localization map is created by combining these weighted maps, which highlights the image regions most responsible for the network's decision. This mechanism provides an interpretable link between the abstract class score and the original visual patterns present in the handwritten word image.

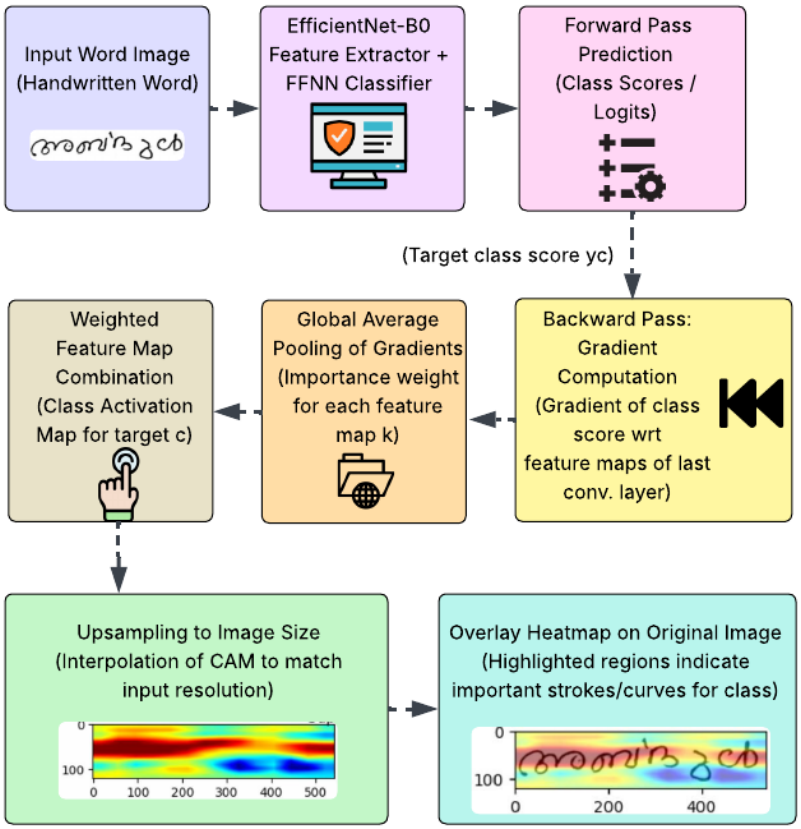

Figure 5.4 Grad-CAM explainability block diagram

Formally, for a target class c, the weight importance α_k^c associated with the k^{th} feature map is given by $\alpha_k^c = \frac{1}{Z_c} \sum_i \sum_j \frac{\partial y^c}{\partial A_{ij}^k}$, in this y^c denotes the score of class c, A_{ij}^k is the activation at spatial location (i, j) of the k^{th} feature map, and Z_c is the normalization constant, typically equal to the number of spatial positions present in the feature map. The weight α_k^c essentially captures the contribution of the k^{th} feature map towards the prediction of class c. Once the weights are obtained, they

are combined with the feature maps to construct the class-specific activation map. It can be defined as $L^c_{Grad-CAM} = ReLU\left(\sum_k \alpha^c_k A^k\right)$. The class activation map generated by Grad-CAM is further improved with the ReLU function. It serves as a filter, storing activations that contribute positively to predicting the chosen class. It ignores signals which dampen or oppose the decision. This selective retention ensures that the visualisation focuses on supportive rather negative evidence (ie, noise or misleading cues). After the refined activation map is produced, it is resized to the resolution of the input image. When applied on top of the handwritten word image, a heatmap is generated which shows visually where in the input space left the most important regions in determining the model classification.

In application, Grad-CAM gives human- readable explanations of decision making on the proposed hybrid EfficientNet-B0 and FFNN model. For instance, when recognizing handwritten Malayalam words, the heatmaps generated by Grad-CAM usually focus on the crucial character strokes or particular patterns in the word image that help to separate one word from another. This is a proof that the model is looking at semantically significant regions. It also helps diagnosing cases of misclassification by exposing whether the model's attention has been redirected to irrelevant or noisy regions. Therefore, Grad-CAM is a crucial tool to make sure that the recognition pipeline is not only accurate, but also transparent, interpretable and trustable in realistic real-time applications. Figure 5.5 shows the results by overlaying the original image on the heat map.

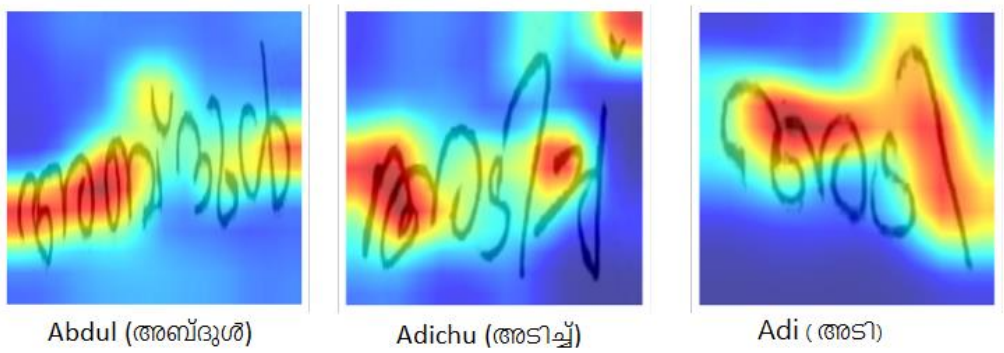

Figure 5.5 Grad-CAM Results

5.5 METHODOLOGY FOR DOCUMENT RETRIEVAL SYSTEM

The retrieval system of documents developed in this chapter is intended as an extension of the handwritten word recognition system. After recognizing the handwritten Malayalam FISs, word sequences that are obtained from the recognition pipeline are converted into structural text representations. Converted to digital text, these documents can be processed computationally and searched semantically.

The retrieval framework is organized into three main stages. In the first stage, each FIS undergoes preprocessing and segmentation, ensuring that handwritten content is cleaned, segmented into words, and accurately transcribed by the EfficientNet-based recognition model. Then in the second stage, the recognized words are normalized and represented numerically using Latent Dirichlet Allocation (LDA) [118], which helps to identify the main theme in the texts. This step enables the system to identify latent semantic patterns across documents rather than relying solely on keyword matching. Finally, in the third stage, user queries are processed and matched against the topic representations of documents, allowing the system to return a ranked set of FISs most relevant to the query.

Through this integration of recognition and retrieval, the system advances beyond simple transcription, developing into a functional tool that assists in navigating large collections of handwritten FISs. This addresses the challenge of searching through handwritten archives.

5.5.1 Document Corpus Preparation

A well-organized collection of documents is always a foundation stone for any text-based retrieval system. In this framework we use a repository containing 50 FIS documents. It is derived from scanned handwritten images with word-level recognition. Let the document set be represented as in Eq. 5.1.

$$Dc = \{d_1, d_2, \dots, d_{50}\} \tag{5.1}$$

Where d_i represents the i^{th} FIS document in the collection. After recognition, each document is defined as a sequence of recognized Malayalam words. To facilitate

effective retrieval, each document is preprocessed and tokenized, converting the raw text into an ordered set of words as in Eq. 5.2.

$$d_i = \{w_{i1}, w_{i2}, \dots, w_{in_i}\} \tag{5.2}$$

Where w_{ij} denotes the j^{th}word in document d_i and in_i is the total number of words in the j^{th} document. This tokenization guarantees that we have already decomposed the text into meaningful lexical units for representation and similarity computation later on. The tokenized document set is referred to as structured corpus given by Eq. 5.3.

$$Cp = \{d_1, d_2, \dots, d_{50}\} \tag{5.3}$$

This corpus serves as the preprocessed input to topic modeling and term weighting in the document retrieval module. Topic modeling approaches are highly sensitive to vocabulary consistency [119].

5.5.2 Dictionary Construction and Bag-of-Words (BoW) Representation

Following tokenization and preprocessing, the next stage of the proposed document retrieval framework is to create a dictionary constructed using the Gensim library [120] and transform documents into a numerical representation suitable for statistical modeling. A dictionary is constructed by counting frequency of each distinct word in Malayalam appearing throughout the collection. This dictionary serves as a global vocabulary that maps each distinct word to a unique integer index. From the structured document corpus Cp, a dictionary V is constructed as shown in Eq. 5.4.

$$V = \{w_1, w_2, \dots, w_{|v|}\} \tag{5.4}$$

Where $|v|$ represents the size of the vocabulary. In the proposed system, the vocabulary so obtained consists of 215 distinct Malayalam words that reflect the lexical variety found in those FIS documents. In this dictionary, we represent each document as a Bag-of-Words (BoW) vector. The BoW model views a document as an unordered collection of word occurrences, and disregards the grammar structure

and word order in the process of keeping term frequency information. Formally, the BoW representation of the i-th document is defined as shown in Eq. 5.5.

$$d_i = \{(w_j, f_{ij}) \mid w_j \in V\} \tag{5.5}$$

Where f_{ij} denotes the frequency of word w_j in document d_i. Words that do not occur in a document have a frequency of zero and are omitted from the representation.

This sparse vector representation enables efficient storage and computation and forms the foundation for subsequent modeling stages. The BoW vectors are used as input for LDA to discover latent topics and for TF–IDF weighting to compute term importance during similarity-based document retrieval.

5.5.3 TF-IDF Weighting Scheme

To reduce the dominance of frequently occurring narrative words in FIS documents, such as "ഞാൻ", "ഇന്ന്", and "മണി", a TF–IDF weighting scheme is applied to the BoW representation of the corpus. FIS documents typically contain repetitive first-person narration and temporal expressions that do not contribute significantly to distinguishing one case from another. TF–IDF effectively addresses this issue by down-weighting such commonly occurring terms and assigning higher importance to discriminative words relevant to the nature of the complaint [121]. For a given word w in a document d, the TF–IDF weight is computed as shown in Eq. (5.6)

$$TF - IDF(w, d) = TF(w, d) \times log\left(\frac{N}{DF(w)}\right) \tag{5.6}$$

Where $TF(w, d)$ denotes the frequency of each word w in document d, N represents the total number of documents present in the corpus, and $DF(w)$ indicates the number of documents in which the word w appears. This formulation ensures that words that occur in many FISs receive lower weights, while words that occur in fewer documents receive higher weights.

By applying TF–IDF weighting, the influence of generic narration terms is significantly reduced. In contrast, crime-specific and semantically informative terms such as "കളവ്" (theft), "മോഷണം" (robbery), and "അപകടം" (accident) are emphasized. This weighted representation improves the quality of similarity computation and document ranking in the subsequent retrieval stage by focusing on content that is more relevant for distinguishing between different FIS cases.

5.5.4 Topic Modeling using LDA

To uncover latent semantic patterns within the recognized FIS document corpus, LDA is employed as an unsupervised topic modeling technique. LDA is particularly well-suited for legal and administrative documents such as FIS, where thematic structures are not explicitly labeled and are instead embedded within recurring narrative expressions and procedural language. The input to the LDA model is the Bag-of-Words representation obtained after dictionary construction. LDA treats each document as a mixture of hidden topics, and the topic is modeled as a probability distribution over the vocabulary.

In this framework, the number of topics is set to three. This choice is motivated by the relatively small corpus size and the need to maintain interpretability of the learned topics. The LDA model is trained using multiple passes over the corpus to ensure stable estimation of both topic–word and document–topic distributions. Each learned topic can be formally expressed as shown in Eq. (5.7)

$$Topic_k = \{(w_1, p_1), (w_2, p_2), \dots\} \tag{5.7}$$

Where w_j denotes a word in the dictionary and p_j = P(w_j | $Topic_k$), represents the probability of that word under the k^{th} topic. After training, each document is associated with a topic probability vector, and the dominant topic is identified as the one with the highest posterior probability. This dominant topic assignment is subsequently used as a semantic filtering mechanism during document retrieval, effectively reducing the search space before applying fine-grained similarity ranking using TF–IDF and cosine similarity. With topic modeling before similarity computation, the model successfully narrows down the search space here (and

presumably provides better retrieval relevance), in particular when there exist considerable lexical overlap between unrelated FIS documents.

5.5.5 Document Topic Assignment

Following LDA model training, each FIS document in the corpus is characterized by a document–topic probability distribution, which quantifies the association between the document and each latent topic. For a given document d_i, LDA generates a set of posterior probabilities as shown in Eq. (5.8)

$$\{(k, P(k \mid d_i))\} \tag{5.8}$$

In order to assign a single representative topic to each document, we select the dominant topic which is the one that has the highest posterior probability. Formally, the topic assignment function is defined as shown in Eq. (5.9)

$$Topic(d_i) = arg^{max}_{k} P(k \mid d_i) \tag{5.9}$$

This dominant topic is the most important latent semantic structure in the content of document. When assigning FIS documents to its most prevalent topic, the corpus is subdivided into topic clusters so that the documents in each cluster have similar narrative style and context status. This theme-level categorization does not force a strict categorization of the type of crime, but rather it captures similar patterns within reports such as writing style, legal terminology and reference to context often present in FIS narratives.

5.5.6 Query Processing and Topic Identification

The user-given Malayalam query (e.g.“ഏലത്തൂർ നടന്ന കളവ്”) is analyzed to recognize its semantic meaning and the same is used for retrieving documents. The query is first tokenized, breaking the input text into individual Malayalam word tokens. The pre-processing on the query is identical to that of the document corpous. These tokens are then matched to words in the dictionary which was created when preparing the corpus, and then using the BoW model, each word is given a corresponding integer value - ID that ensures that there exists an equivalent

vector present on the FIS documents. The BoW of the query is input to the pre-trained LDA model to infer its topic distribution (i.e., posterior probability over each latent topics conditioned on the query). Specifically, the most dominant topic of the query is extracted as in Eq. (5.10)

$$Topic(q) = arg^{max}_{k} P(k \mid q) \tag{5.10}$$

Where $P(k \mid q)$ is the probability that query falls under topic p. This dominant theme is the major semantic context of the query. This way, the retrieval is limited to FIS documents from that topic. It guaranteed semantic consistency in the query and candidate document. This topic-specific constraint greatly reduces the search space, enhancing retrieval relevance by eliminating comparison with semantically irrelevant documents, as such holdback of sentence using semantic non-related words mostly a necessity in narrative type FIS text which are rich in common procedural terms.

5.5.7 Topic-based Document Filtering

After identifying the dominant topic of the input query, the document retrieval process is constrained through topic-based filtering to ensure semantic relevance. Each FIS document in the corpus has already been assigned a dominant topic based on its maximum posterior topic probability obtained from the trained LDA model. Let the dominant topic of the query be denoted as $Topic(q)$. Only those documents d_i for which $Topic(d_i) = Topic(q)$ are selected to form a reduced candidate set for retrieval. This filtered document subset contains FIS documents that share similar latent semantic characteristics with the query. By excluding documents belonging to unrelated topics, the system significantly reduces the number of comparisons required during similarity computation. This topic-based filtering step not only improves computational efficiency but also minimizes false matches arising from standard procedural or narrative terms that frequently appear across FIS documents. As a result, subsequent similarity ranking is performed on a semantically coherent document set, yielding more accurate and meaningful retrieval results.

5.5.8 Calculating Similarity Based on TF-IDF and Cosine Similarity

After topic-based filtering, similarity computation ranks the remaining FIS documents by their relevance to the input query. Both the query and the filtered documents are represented in the TF–IDF vector space, where each dimension corresponds to a term in the dictionary, and the associated weight reflects the term's importance within the document and across the corpus. To efficiently compute similarity scores between the query vector and multiple document vectors, a similarity index is constructed using the TF–IDF representations of the filtered document set. This index enables fast comparison by computing cosine similarity between sparse vectors [122]. Cosine similarity measures the angular similarity between two vectors and is defined as shown in Eq. (5.11)

$$\mathrm{sim}(\mathrm{d}, \mathrm{q}) = \frac{\mathrm{d} \cdot \mathrm{q}}{\| \mathrm{d} \| \| \mathrm{q} \|} \tag{5.11}$$

Where, d denotes the TF–IDF vector of a document and q denotes the TF–IDF vector of the query. This measure captures the overlap between the weighted term distributions of the query and each document, regardless of document length. The similarity index used in this work efficiently implements cosine similarity over sparse TF–IDF vectors without introducing a separate similarity metric.

Once similarity scores are computed for all filtered documents, the documents are ranked in descending order of their similarity values. Documents with higher cosine similarity scores exhibit greater TF–IDF overlap with the query and are therefore considered more relevant. The top-ranked documents are returned as the final retrieval result, ensuring that the system presents FIS documents that are both semantically aligned through topic modeling and lexically relevant through term-weighted similarity matching.

5.6 RESULTS AND DISCUSSION

The proposed document retrieval system has been verified through combining the EfficientNet-B0 with LDA for the word recognition based search model. Recognition model enables successful transformation of handwritten Malayalam

FISs into search able text. The Grad-CAM explainability presented the visual evidence of model focus during classification, which enhanced trust and transparency in their recognition pipeline. The recognized text was parsed and indexed utilizing LDA in order to perform semantic search of corresponding documents.

The LDA was trained with three topics to model the latent semantic structure in the FIS corpus. The learned topic–word distributions are presented in Table 5.2.

Table 5.2 Extracted LDA Topics and Representative Keywords

Topic ID	High-Probability Words (Top Terms)	Interpretation
Topic 0	ഞാൻ, ഇന്ന്, പിടിച്ചു, കോളേജ്, മണിക്ക്, മെഡിക്കൽ, നിന്നും	Incident descriptions with temporal references and medical or institutional context
Topic 1	ഞാൻ, എനിക്ക്, എന്നെ, സ്റ്റേഷനിൽ, സമയത്ത്, മണി, കോഴിക്കോട്	Police station narration and complaint registration
Topic 2	ഞാൻ, മൊഴി, വായിച്ചു, നോക്കി, സ്റ്റേഷനിൽ, എന്നയാൾ	Witness statements and verification procedures

As shown in this table, the extracted topics are mostly associated with procedural and narrative aspects rather than direct crime labeling. This is in line with the formal structure of FIS documents, where identical phrasing occurs multiple times over cases.

This included tokenization, BoW conversion and topic inference based on LDA of the Malayalam query "എലത്തൂർ നടന്ന കളവ്". The most prevalent topic for the query is assigned as Topic 1 and it presents semantically similar words to that of police station description and formal complaint filing. After filtering the topics, the related FIS documents were ranked based on TF–IDF weighting and cosine similarity. The top-ranked documents and their similarity scores are shown in Table 5.3.

Table 5.3 Top five retrieved documents for the query with similarity score

Rank	Retrieved Document	Similarity Score
1	Document 33	0.3240
2	Document 42	0.2517
3	Document 15	0.2298
4	Document 9	0.1454
5	Document 37	0.1185

The above results presents the top five retrieved documents for the query, along with representative terms and similarity scores. The tables results prove that the topic modelling oimproves relevance filtering, and that similarity scores justify document positioning.

The results illustrate three main points of this chapter:

- Recognition Accuracy - EfficientNet-B0 with FFNN provided strong recognition performance (Accuracy: 98.68%, F1-score: 0.9869), guaranteeing the reliability at word-level of digitizing handwritten data.
- Explainability - Grad-CAM visualizations showed that the recognition model also focused on meaningful stroke regions, which enhanced interpretability of the model.
- Retrieval Performance - The TF-IDF-LDA-based retrieval method effectively ranked documents that contained semantic-matching words (though queries were not exactly matched with document texts).

In summary, the system connects the two sub areas of handwriting recognition and retrieval from handwritten databases providing a pragmatic solution to convert raw unstructured Malayalam FISs into a searchable knowledge. Such integration can significantly improve practical application for crime analysis and legal documentation related operations.

5.7 SUMMARY

In this chapter, a document retrieval system that combines handwriting word recognition and semantic retrieval is described. An EfficientNet-B0 based model was utilized as a feature extractor and FFNN for classification to ensure high accuracy with low computational cost and convenient for low resource compatibility.

The chapter also highlighted the explainability of the model by providing visualization of significant regions in word images for recognition using Grad-CAM to make the model more interpretable. For document retrieval, recognized words were used to construct a TF-IDF weighted dictionary, and LDA was utilized to discover the hidden topic behind the documents. User queries were converted, through transformation, into the same feature space and similarities (cosine similarity) were computed between pairs to retrieve documents.

Experiments showed that the system has high recognition accuracy and efficient retrieval performance, with explainable and interpretable results. In conclusion, this chapter provides a solid and efficient solution for the combination of handwritten word recognition and semantic document retrieval with a good accuracy-efficiency and low resource friendly device compatibility trade-off.

6. CONCLUSION

6.1 THESIS SUMMARY

This study performed handwritten word recognition for the Malayalam language and expand it towards a document retrieval model with hybrid deep learning methods. The thesis is organized around three principal contributions. The creation of handwritten word dataset, development of an effective recognition model, and the deployment on resource constraint retrieval system.

The work first establishes a curated dataset of handwritten Malayalam words. The handwritten source materials were obtained from real police FISs and supplemented with additional handwritten samples to ensure adequate representation of the script's natural variability. Each word image was carefully segmented, labelled, and validated, resulting in a balanced dataset suitable for training and evaluation. The dataset contains samples that capture variations in writing style, stroke dynamics, and spacing, thereby reflecting real-world conditions encountered in administrative and legal documents.

The second contribution focuses on designing a high-performing handwritten word recognition framework. A transformer-based feature extractor was employed to capture both global and fine-grained patterns in handwritten words, while an attention-enhanced feed forward neural network was used for classification. Comprehensive experiments demonstrated that this hybrid architecture provides strong generalization across diverse handwriting styles, achieving high accuracy and stable performance across several evaluation metrics.

The third component of the thesis extends the recognition capability to a document retrieval application. Preprocessed and segmented words from handwritten documents were first recognized using a lightweight EfficientNet-based model suited for deployment on edge devices. The recognized tokens were then indexed using TF-IDF representations and an LDA model to support semantic matching.

This enabled users to input a Malayalam text query and retrieve the most relevant handwritten documents based on topic similarity. Explainability tools such as Grad-CAM were also incorporated to analyze model behavior and support transparent decision-making.

In combination, the three components form a complete pipeline that spans data acquisition, recognition, and retrieval. The study contributes a new dataset, MHWD, a high-accuracy recognition model, TrA-MHWR, and a retrieval system designed for a real-world handwritten document retrieval task. Thereby advancing the field of handwritten Malayalam document processing and opening new research and deployment directions.

6.2 CONTRIBUTION OF THE THESIS

The study enhances HR by addressing the challenges faced by low-resource, complex-script languages like Malayalam. The main contributions of this study are outlined below:

- MHWD Creation: MHWD has been developed to support research in recognizing handwritten Malayalam words. Since the dataset has not yet been publicly released, it is available from the authors upon request for research purposes. The dataset includes 321 unique word classes containing 150 handwritten samples in each class, resulting in 48,150 word images. These samples have been sourced from real-world police FIS and balanced by collecting handwriting from individuals aged 10 to 65. MHWD fills a gap in publicly available resources for Malayalam handwritten word recognition and is available as a benchmark dataset for future researchers. In this study, the dataset is intended to be hosted on an open-source platform in order to facilitate further investigations.

- Development of TrA-MHWR Model for Handwritten Word Recognition: The research proposes the Transformer-based Malayalam handwritten word recognition deep learning model tailored specifically to the complex structure of the Malayalam script. TrA-MHWR adopts transformer with

residual connections for recognition, which is able to learn local fine details and global word-level structures effectively based on attention mechanisms.

- Performance and Generalization Analysis: The TrA-MHWR model is evaluated on the MHWD dataset and compared with state-of-the-art CMATERdb2.1.2 standard benchmark dataset. One option would be to test robustness and generalization of the proposed features to other Indic script, e.g. Bangla from other Indic script [123]. The findings verify the model's generalizablity over varying handwriting styles, hence indicating that it is a robust technique for low-resource script recognition

- Document Retrieval System Implementation: To demonstrate the practical ability of our recognition system, a handwritten document retrieval system is developed. The system allows searching for documents in dataset of handwritten pages. In this system, a light-weight EfficientNet-based model is used for recognition purposes, making it suitable with constrained computational recourses. This makes it possible to use the system on devices with limited resources, allowing its real-world usability. The explainability of the EfficientNet-based model employed in the document retrieval system is also investigated.

- A Framework for Low Resource Scripts: The end-to-end methodology including the procedure of dataset creation, TrA-MHWR model building with document retrieval systems incorporated offers a framework which can be generalized to any language. This framework can be extended to other low-resource, complex-script languages, offering valuable guidance for future research in document image analysis and handwritten text recognition.

7. RECOMMENDATIONS FOR FUTURE RESEARCH

7.1 INTRODUCTION

The study has made significant contribution to the domain of Malayalam handwritten word recognition. A novel hybrid deep learning framework based on transformers named TrA-MHWR has been proposed. The development of a handwritten word dataset, MHWD, and the document retrieval system shows that the framework can be deployed in real-world environments. The performance resulted using the proposed approach in addressing various challenges of Malayalam language like complex character shapes, compound characters and handwriting variants substantiate its effectiveness.

This study also suggests a few future directions for investigations. The handwritten word recognition in Malayalam and document retrieval using this recognition system is a developing field of research. The results of this paper may be used as a solid baseline for future research aiming at improving and developing the methods advanced in this manuscript. Future efforts might include improving the scalability and generalizability of recognition model by dealing with dataset limitations diversity-wise, and extending the retrieval framework to more realistic, large-scale scenarios. More sophisticated generalization techniques, interpretability and user-centered design may improve both the accuracy of recognition and the usability of the system.

One of the most problematic steps of this pipeline was at the initial level: both document image preprocessing and segmentation. Most of the gathered handwritten files were degraded which suffered from noise, ink-spreading, faded strokes and non-alignment. In most cases even partly overlapping lines and words seemed connected, leaving segmentation highly challenging. Pre-processing, morphological operations and word segmentation were used to resolve these problems in this work. The outcome showed that noise reduction and segmentation are still major

bottlenecks. In future work these need to be addressed through more sophisticated denoising techniques, adaptive binarization techniques and robust segmentation algorithms developed specifically for Malayalam handwritten script found in documents like FIS. Improvements in these primitive stages could directly facilitate the quality and diversity of input data, which would contribute to the enhancement of recognition precision and system reliability.

Promising future directions in research are summarized in the sub-section. This guidelines are serving to orient future research towards unsolved challenges and to undisclosed new avenues in the analysis of handwritten Malayalam manuscripts. The expansion of the existing Malayalam handwriting datasets, improvements to the methodology employed, and cross-linguistic applicability; as well as practical deployment considerations, will ensure that future efforts increase not only recognition accuracy but also contribute to strengthening the wider framework of intelligent document retrieval systems.

7.2 DATASET EXPANSION AND DIVERSITY

The dataset created in this work, named MHWD, is important for recognition of handwritten Malayalam words. MHWD is utilized to compare the performance of the developed hybrid recognition model, TrA-MHWR. Its current version includes 321 word classes with an equal amount of samples. This is the regulated structure for modeling and testing recognition model. However, it limits the usability of the dataset for large-scale usage. A useful recognizer should therefore at least be suitable for a significantly larger vocabulary it could cover Malayalam usages in the real world context.

Thus, further research needs to be dedicated to increasing the dataset that should contain not only more frequent vocabulary, but also rare and less common words, both general (words of daily life) and specialized vocabulary from domains such as law, medicine and administration. This extension would allow recognition models to be used across a range of different application domains such as converting government documents to digital format, archiving preservation and domain-specific document retrieval.

An additional dimension of dataset enrichment is greater handwriting variability. This variation is due to factors such as age, education level, regional and social class dialects, or personal writing habits. A richer dataset must be compiled by sampling among a wider and more diverse audience base to cover the full array of variations in handwritten styles that may typically be encountered. This would strengthen the recognition models and alleviate over-fitting to small set of writing styles.

In summary, an important suggestion for future work is that there should be a specific benchmark dataset available to the research community constituting standard benchmark data with clear-cut annotations for Malayalam handwriting recognition. Such a tool would offer an ideal testing bed for novel architectures and approaches. This will enable future evaluators to quantify improvements in a transparent and reproducible way. In conclusion, enhancing the dataset in terms of vocabulary size and creating an open benchmark will be a key to improve scope and generalization capability of the Malayalam handwritten word recognition systems. Such an endeavor would enhance the recognition accuracy and reinforce the retrieval framework that can be applied to bigger and realistic document collections.

7.3 SEGMENTATION AND PREPROCESSING IMPROVEMENTS

The pre-processing and segmentation of the novel recognition pipeline was a critical challenge in this study. The scanned images of the handwritten Malayalam documents used for this study were noisy. The document images have experienced degradation caused by ink fading, paper aging, smudge and noise added in the scanning process. Some parts of characters were dropped or superimposed and spaces were not always uniform. These problems made it difficult to segment words in the document image. In the current study, noise reduction, morphological operations and contour-based segmentation were used to reduce these problems. However, segmentation is still a challenging step in some document images because of the problems mentioned above.

Further research should therefore focus more on the development of advanced pre-processing and segmentation techniques for Malayalam script. Advanced denoise

methods, especially deep learning based ones, could remove noise selectively and maintain the crucial strokes to ensure the clarity of input images.

The segmentation itself is even more difficult. Due to the cursive nature of Malayalam handwriting, you are forced to combine characters and words and it is hard to figure out where one ends and another begins. In these instances, traditional projection profiles or connected component approaches may be less effective at producing a faithful segmentation. Other approaches like graph-based segmentation, contour learning, or attention-based structure segmentation have to be explored in future. Another promising direction would be to use Generative Adversarial Networks (GANs) or image restoration methods for reconstructing distorted or occluded characters prior to segmentation, so that the input for the downstream recognition is of high quality.

Bettering the segmentation and pre-processing is one of the basic needs to promote a handwritten Malayalam word recognition system. Any enhancement at this level contributes to the robustness of the recognition models by giving a more accurate input. Accordingly, future work can consider segmentation and pre-processing as integral part of recognition pipeline while also pursuing more sophisticated methods that integrate classical image processing with contemporary machine learning.

7.4 WORD RECOGNITION WITHOUT LEXICAL CONSTRAINTS

The MHWR system that was developed in this work had been designed specifically for a given word class size of 321 classes of Malayalam words. This pre-determined choice was made on methodological grounds to keep the training and evaluation consistent, enabling the recognition system to learn different word-level representations. Results have shown that the suggested method worked well with a high recognition rate in the limited vocabulary. However, this poses an elementary bottleneck for scaling it up and designing practical systems. In reality, words in a handwritten document are not restricted to constrained lexicon, they could be any type of word such as name or number or abbreviation, rarely occurring words that might not even available in training data. Therefore, how to generalize from

recognition of fixed lexicons into open vocabulary is an important future research direction.

Character or sub word level recognition is one way to achieve this goal. In that, the recognition model would be trained to recognize basic character elements or some group of characters in the word and not a whole word. This would enable the system to generate out of vocabulary words. For the Malayalam script, with its complex combined and compound characters, such methods require methods that can intelligently split sub segments of a character into its constituent parts which taken together forms a whole element or where those same smaller units join together to become part of one larger unit in the next level linguistic analysis.

Using sub word or phonetic representations might help generalize recognition to new words without any retraining of the whole system. The open-vocabulary approach combines recognition and retrieval mechanisms. Recognition model returns an initial hypothesis which may further be refined by searching a bigger lexicon or document repository to find the best matching information. The recognition-retrieval schemes can also properly work in reliability of understanding each uncertain or unclear case with such contextual information by which the final decision is made. A partially recognized word provided by the proposed model may be compared with words in a word list using techniques referred to as similarity metrics including edit distance or phonetic alignment. Accordingly, the recognition accuracy of even novel words can be improved.

Recognition of words beyond fixed lexicons is a crucial stage to make the handwritten Malayalam word recognition systems more realistic and scalable. Future research can enable models to handle a much broader range of handwritten inputs by shifting from rigid lexicon based recognition to open and adaptive frameworks. This will enhance recognition performance and expand the scope of the document retrieval system.

7.5 POST-RECOGNITION ERROR CORRECTION

The proposed hybrid deep neural network model achieved good recognition accuracy. However, occasional misclassifications cannot be avoided in the handwritten word recognition task, especially for degraded documents or ambiguous writing style. Post-recognition correction is improving the output by applications of verification, or correction processes. The error correction process following recognition also improves the robustness of a model. In document digitization or retrieval, small recognization errors can also lead to semantic differences and may even result in no matches during retrieval. Thus, the efficiency of the system could be enhanced by applying a strong error correction module.

The most common mistakes made by a handwritten-word recognizer can be attributed to the similarity of some Malayalam hand-written letters and partial strokes segmentation. Such errors will be caught and corrected through smart correction modules in the future. One possible strategy would be dictionary-based correction, by matching the recognised output with an extensive Malayalam lexicon. The system can then use similarity calculations like edit distance or phonetic match to return the most likely correct word(s) that are close by value from the model output. This technique can greatly reduce the number of recognition errors that occur, particularly when a predicted word is quite similar to those within the valid dictionary.

Another potential approach lies in detecting and learning the common error patterns of the recognition model. An independent correction module can be designed to predict and correct the common errors through the analysis of misclassifications in training and testing phases. For instance, if certain characters are frequently repetitive and visually indistinguishable, then the correction system can tend to favor substitutions that make more linguistic or contextual sense. This type of error-aware correction makes the system robust for various handwriting styles.

Lightweight neural models can also contribute to correcting along with rule and dictionary based techniques. Such models will analyse sequences of predicted characters or words and then produce corrected versions by applying knowledge

about language which these models have derived. Such models can be added after the recognition step to enhance performance without significant latency overhead. A mixed solution of the two, where dictionaries act as a filter and neural passes serve as correction seems to provide best trade-offs between accuracy and speed.

In summary, adding a post-recognition correction step can greatly increase the robustness of handwritten Malayalam word recognition and document retrieval systems. It guarantees that the final intended information is accurate and meaningful, even though there might exist some slight recognition errors. it is expected that future works will have to concentrate on the design of correction mechanisms that meld lexical, contextual and learned processors ensuring the observed level of precision and robustness whenever real documents are processed.

7.6 CONTEXTUAL DOCUMENT RETRIEVAL ENHANCEMENTS

The document retrieval system proposed in his research used LDA modelling was used to retrieve the Malayalam documents. By learning the thematic distribution over known words, it clusters documents into coherent topics and allows for sentence-level search. This permits users to write contextual queries, e.g., "കോഴിക്കോട് നടന്ന ആത്മഹത്യ" ("suicide in Kozhikode") and capture documents with the thematic content similar to that query.

This component will be enhanced in the future by incorporating semantic embedding models and sophisticated, deep context representations. Methodologies that are based on embeddings, such as fastText, Word2Vec, or transformer-based representations can learn intermediate subtle word relationships by mapping them in a high- dimensional vector space. Adopting this would enable the retrieval system to understanding the explicit intention of a query than the surface word distributions used at present.

Another useful direction is to optimize query expansion and contextual disambiguation methods. That is, if a user searches for something like "കോഴിക്കോട് നടന്ന ആത്മഹത്യ" (suicide in kozhikode) the system should be able to adaptively

include related searches such as "മരണ റിപ്പോർട്ട്" (death report).. These mechanisms would enable the system to be robust towards linguistic diversit and contextual variability, which are routinely encountered in handwritten as well as real-world Malayalam text.

Through increasing the capacity for understanding of queries in context and by using richer retrieval strategies, future systems could progress towards a more intelligent and human-like perception of documents. These will not only enhance retrieval precision but would also facilitate the building of scalable and adaptive information systems on extensive sets of handwritten Malayalam documents.

7.7 CONCLUSION

In this chapter, we provide implications of this study for future research and new directions with which these can be integrated. Some of the key directions proposed are data enrichment, better segmentation and pre-processing, recognition when vocabulary is not fixed or further expanded post-recognition and error correction strategy which should be robust (context dependent) and context aware retrieval. Of these, the enhancement of segmentation and preprocessing is a difficult one as they suffer from noise and degradation contained in real-word handwritten documents. Dealing with such limitations will further improve the accuracy of recognition and the reliability of retrieval.

Together, these suggestions provide a roadmap for developing more scalable, adaptive and usable systems of handwritten Malayalam recognition and retrieval. Future work in this line can help to make a contribution for wider purpose including linguistic heritage generation, increasing availability for handwritten archives and enabling intelligent document processing across various practical domains.

REFERENCES

[1] R. Plamondon and S. N. Srihari, "Online and off-line handwriting recognition: a comprehensive survey," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 22, no. 1, pp. 63–84, 2000, doi: 10.1109/34.824821.

[2] P. Ghadekar, O. Khanvilkar, J. Kharat, K. Joshi, T. Kulkarni, and Y. Kulkarni, "Automating the grading of handwritten examinations through the integration of optical character recognition and machine learning algorithms," *J. Integr. Sci. Technol.*, vol. 13, no. 6, pp. 1128–1128, 2025, doi: 10.62110/sciencein.jist. 2025.v13.1128.

[3] C. Garrido-Munoz, A. Rios-Vila, and J. Calvo-Zaragoza, "Handwritten Text Recognition: A Survey," Feb. 12, 2025, *arXiv*: arXiv:2502.08417. doi: 10.48550/arXiv.2502.08417.

[4] M. Jabde, C. H. Patil, A. D. Vibhute, and J. R. Saini, "A systematic review of multilingual numeral recognition systems," *Artif. Intell. Rev.*, vol. 58, no. 4, p. 106, Jan. 2025, doi: 10.1007/s10462-025-11105-0.

[5] S. Singh, A. Sharma, and V. K. Chauhan, "Indic script family and its offline handwriting recognition for characters/digits and words: a comprehensive survey," *Artif. Intell. Rev.*, vol. 56, no. S3, pp. 3003–3055, Dec. 2023, doi: 10.1007/s10462-023-10597-y.

[6] S. Madhvanath and V. Govindaraju, "The role of holistic paradigms in handwritten word recognition," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 23, no. 2, pp. 149–164, 2001, doi: 10.1109/34.908966.

[7] B. Jose and P. K. P., "Classification of handwritten Malayalam characters using a HOG-DCNN model with multiview augmentation and inference fusion," *Multimed. Tools Appl.*, vol. 83, no. 7, pp. 19969–19980, Jul. 2023, doi: 10.1007/s11042-023-16154-7.

[8] Loknath Behera IPS, State Police Chief, "Kerala Police Standard Operating Procedure." [Online]. Available: https://keralapolice.gov.in/storage/pages/ custom/ckFiles/file/9BKejc9JqXhlZC76l8TizoeGgLl1y3kVRQU9OEQD.pdf

[9] M. Boualam, Y. Elfakir, G. Khaissidi, and M. Mrabti, "Arabic Handwriting Word Recognition Based on Convolutional Recurrent Neural Network," in *WITS 2020*, vol. 745, S. Bennani, Y. Lakhrissi, G. Khaissidi, A. Mansouri, and Y. Khamlichi, Eds., in Lecture Notes in Electrical Engineering, vol. 745. , Singapore: Springer Singapore, 2022, pp. 877–885. doi: 10.1007/978-981-33-6893-4_79.

[10] J. Michael, R. Labahn, T. Grüning, and J. Zöllner, "Evaluating sequence-to-sequence models for handwritten text recognition," in *2019 International Conference on Document Analysis and Recognition (ICDAR)*, IEEE, 2019, pp. 1286–1293. doi: 10.1109/ICDAR.2019.00208.

[11] Y. Li, D. Chen, T. Tang, and X. Shen, "HTR-VT: Handwritten text recognition with vision transformer," *Pattern Recognit.*, vol. 158, p. 110967, 2025, doi: 10.1016/j.patcog.2024.110967.

[12] M. Li *et al.*, "Trocr: Transformer-based optical character recognition with pre-trained models," in *Proceedings of the AAAI conference on artificial intelligence*, 2023, pp. 13094–13102. doi: 10.48550/arXiv.2109.10282.

[13] J. J. Hull, "A database for handwritten text recognition research," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 16, no. 5, pp. 550–554, 1994, doi: 10.1109/34.291440.

[14] U.-V. Marti and H. Bunke, "The IAM-database: an English sentence database for offline handwriting recognition," *Int. J. Doc. Anal. Recognit.*, vol. 5, no. 1, pp. 39–46, Nov. 2002, doi: 10.1007/s100320200071.

[15] T. Bluche, H. Ney, and C. Kermorvant, "Feature extraction with convolutional neural networks for handwritten word recognition," in *2013 12th international conference on document analysis and recognition*, IEEE, 2013, pp. 285–289. doi: 10.1109/ICDAR.2013.64.

[16] "The RIMES database." Accessed: May 18, 2025. [Online]. Available: https://teklia.com/research/rimes-database/

[17] C. Joshi, L. Sorenson, A. Wolfert, M. Clement, J. Price, and K. Buckles, "CENSUS-HWR: a large training dataset for offline handwriting recognition," in *2023 International Conference on Computational Science and Computational Intelligence (CSCI)*, IEEE, 2023, pp. 592–597. doi: 10.1109/CSCI62032.2023.00105.

[18] F. Kleber, S. Fiel, M. Diem, and R. Sablatnig, "Cvl-database: An off-line database for writer retrieval, writer identification and word spotting," in *2013 12th international conference on document analysis and recognition*, IEEE, 2013, pp. 560–564. doi: 10.1109/ICDAR.2013.117.

[19] A. Mondal, K. Tulsyan, and C. V. Jawahar, "Bridging the Gap in Resource for Offline English Handwritten Text Recognition," in *Document Analysis and Recognition - ICDAR 2024*, vol. 14805, E. H. Barney Smith, M. Liwicki, and L. Peng, Eds., in Lecture Notes in Computer Science, vol. 14805. , Cham: Springer Nature Switzerland, 2024, pp. 413–428. doi: 10.1007/978-3-031-70536-6_25.

[20] M. Pechwitz, S. S. Maddouri, V. Märgner, N. Ellouze, and H. Amiri, "IFN/ENIT-database of handwritten Arabic words," in *Proc. of CIFED*, Citeseer, 2002, pp. 127–136. Accessed: May 18, 2025. [Online]. Available: https://www.academia.edu/download/94769098/CIFED_02_ifn-enit-database.pdf

[21] S. A. Mahmoud *et al.*, "Khatt: Arabic offline handwritten text database," in *2012 International conference on frontiers in handwriting recognition*, IEEE, 2012, pp. 449–454. doi: 10.1109/ICFHR.2012.224.

[22] M. E. Hussein, M. Torki, A. Elsallamy, and M. Fayyaz, "AlexU-Word: A New Dataset for Isolated-Word Closed-Vocabulary Offline Arabic Handwriting Recognition," Nov. 17, 2014, *arXiv*: arXiv:1411.4670. doi: 10.48550/arXiv.1411.4670.

[23] M. Saeed *et al.*, “Muharaf: Manuscripts of handwritten arabic dataset for cursive text recognition,” *Adv. Neural Inf. Process. Syst.*, vol. 37, pp. 58525–58538, 2024.

[24] C.-L. Liu, F. Yin, D.-H. Wang, and Q.-F. Wang, “CASIA online and offline Chinese handwriting databases,” in *2011 international conference on document analysis and recognition*, IEEE, 2011, pp. 37–41. doi: 10.1109/ICDAR.2011.17.

[25] T. Su, T. Zhang, and D. Guan, “HIT-MW dataset for offline Chinese handwritten text recognition,” in *Tenth International Workshop on Frontiers in Handwriting Recognition*, Suvisoft, 2006. Accessed: May 19, 2025. [Online]. Available: https://inria.hal.science/inria-00103725/

[26] D. Nurseitov, K. Bostanbekov, D. Kurmankhojayev, A. Alimova, A. Abdallah, and R. Tolegenov, “Handwritten Kazakh and Russian (HKR) database for text recognition,” *Multimed. Tools Appl.*, vol. 80, no. 21–23, pp. 33075–33097, Sep. 2021, doi: 10.1007/s11042-021-11399-6.

[27] A. Yavariabdi, H. Kusetogullari, T. Celik, S. Thummanapally, S. Rijwan, and J. Hall, “Cardis: A swedish historical handwritten character and word dataset,” *IEEE Access*, vol. 10, pp. 55338–55349, 2022, doi: 10.1109/ACCESS.2022.3175197.

[28] S. Gongidi and C. V. Jawahar, “iiit-indic-hw-words: A Dataset for Indic Handwritten Text Recognition,” in *Document Analysis and Recognition – ICDAR 2021*, vol. 12824, J. Lladós, D. Lopresti, and S. Uchida, Eds., in Lecture Notes in Computer Science, vol. 12824. , Cham: Springer International Publishing, 2021, pp. 444–459. doi: 10.1007/978-3-030-86337-1_30.

[29] A. Mondal and C. V. Jawahar, “Unconstrained Camera Captured Indic Offline Handwritten Dataset,” in *Pattern Recognition*, vol. 15319, A. Antonacopoulos, S. Chaudhuri, R. Chellappa, C.-L. Liu, S. Bhattacharya, and U. Pal, Eds., in Lecture Notes in Computer Science, vol. 15319. , Cham: Springer Nature Switzerland, 2025, pp. 333–348. doi: 10.1007/978-3-031-78495-8_21.

[30] M. A. Rahman, N. Tabassum, M. Paul, R. Pal, and M. K. Islam, “Bn-htrd: A benchmark dataset for document level offline bangla handwritten text recognition (htr) and line segmentation,” in *Computer Vision and Image Analysis for Industry 4.0*, Chapman and Hall/CRC, 2023, pp. 1–16. [Online]. Available: https://www.taylorfrancis.com/chapters/edit/10.1201/9781003256106-1/bn-htrd-benchmark-dataset-document-level-offline-bangla-handwritten-text-recognition-htr-line-segmentation-md-ataur-rahman-nazifa-tabassum-mitu-paul-riya-pal-mohammad-khairul-islam

[31] B. Sareen, R. Ahuja, and A. Singh, “A benchmark Gurmukhi offline handwritten dataset of tehsil and sub tehsil names of Punjab,” *SGS-Eng. Sci.*, vol. 1, no. 01, 2021, Accessed: May 19, 2025. [Online]. Available: https://spast.org/techrep/article/view/3139

[32] S. Bhowmik, S. Malakar, R. Sarkar, S. Basu, M. Kundu, and M. Nasipuri, “Off-line Bangla handwritten word recognition: a holistic approach,” *Neural Comput. Appl.*, vol. 31, no. 10, pp. 5783–5798, Oct. 2019, doi: 10.1007/s00521-018-3389-1.

[33] G. Kumar, P. K. Bhatia, and I. Banger, “Analytical review of preprocessing techniques for offline handwritten character recognition,” *Int. J. Adv. Eng. Sci.*, vol. 3, no. 3, pp. 14–22, 2013.

[34] G. Kim and V. Govindaraju, “A lexicon driven approach to handwritten word recognition for real-time applications,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 19, no. 4, pp. 366–379, 1997, doi: 10.1109/34.588017.

[35] J. Dasgupta, K. Bhattacharya, and B. Chanda, “A holistic approach for Off-line handwritten cursive word recognition using directional feature based on Arnold transform,” *Pattern Recognit. Lett.*, vol. 79, pp. 73–79, 2016, doi: 10.1016/j.patrec.2016.05.017.

[36] K. Jayech, M. A. Mahjoub, and N. E. B. Amara, “Arabic handwritten word recognition based on dynamic bayesian network.,” *Int Arab J Inf Technol*, vol. 13, no. 6B, pp. 1024–1031, 2016.

[37] P. P. Roy, A. K. Bhunia, A. Das, P. Dey, and U. Pal, “HMM-based Indic handwritten word recognition using zone segmentation,” *Pattern Recognit.*, vol. 60, pp. 1057–1075, 2016, doi: 10.1016/j.patcog.2016.04.012.

[38] S. Singh, T. Kariveda, J. D. Gupta, and K. Bhattacharya, “Handwritten words recognition for legal amounts of bank cheques in English script,” in *2015 eighth international conference on advances in pattern recognition (ICAPR)*, IEEE, 2015, pp. 1–5. doi: 10.1109/ICAPR.2015.7050716.

[39] M. S. Kadhm and A. K. A. Hassan, “Handwriting word recognition based on SVM classifier,” *Int. J. Adv. Comput. Sci. Appl.*, vol. 1, no. 6, pp. 64–68, 2015.

[40] R. S. Hegadi, P. M. Kamble, A. S. Sherikar, and B. V. Dhandra, “Multiwavelet and connected pixel based feature for handwritten Marathi characters,” in *AIP Conference Proceedings*, AIP Publishing, 2018. doi: 10.1063/1.5047728.

[41] Q. A. A. Nuzaili, S. Z. M. Hashim, F. Saeed, M. S. Khalil, and D. B. Mohamad, “Pixel distribution-based features for offline Arabic handwritten word recognition,” *Int. J. Comput. Vis. Robot.*, vol. 7, no. 1/2, p. 99, 2017, doi: 10.1504/IJCVR.2017.081243.

[42] S. Malakar, M. Ghosh, S. Bhowmik, R. Sarkar, and M. Nasipuri, “A GA based hierarchical feature selection approach for handwritten word recognition,” *Neural Comput. Appl.*, vol. 32, no. 7, pp. 2533–2552, Apr. 2020, doi: 10.1007/s00521-018-3937-8.

[43] M. Stauffer, A. Fischer, and K. Riesen, “A Novel Graph Database for Handwritten Word Images,” in *Structural, Syntactic, and Statistical Pattern Recognition*, vol. 10029, A. Robles-Kelly, M. Loog, B. Biggio, F. Escolano, and R. Wilson, Eds., in Lecture Notes in Computer Science, vol. 10029. , Cham: Springer International Publishing, 2016, pp. 553–563. doi: 10.1007/978-3-319-49055-7_49.

[44] R. Zaghdoudi and H. Seridi, “Combination of Multiple Classifiers for Off-Line Handwritten Arabic Word Recognition.,” *Int. Arab J. Inf. Technol. IAJIT*, vol. 14, no. 5, 2017, Accessed: May 22, 2025. [Online]. Available: https://iajit.org/PDF/vol%2014,%20no.%205%20sep/9553.pdf

[45] S. Sawant and S. Baji, “Handwritten character and word recognition using their geometrical features through neural networks,” *Int. J. Appl. Innov. Eng. Manag. IJAIEM*, vol. 5, 2016, Accessed: May 22, 2025. [Online]. Available: https://www.academia.edu/download/47990289/IJAIEM-2016-07-18-15.pdf

[46] M. S. Patel, S. L. Reddy, and A. J. Naik, “An Efficient Way of Handwritten English Word Recognition,” in *Proceedings of the 3rd International Conference on Frontiers of Intelligent Computing: Theory and Applications (FICTA) 2014*, vol. 328, S. C. Satapathy, B. N. Biswal, S. K. Udgata, and J. K. Mandal, Eds., in Advances in Intelligent Systems and Computing, vol. 328. , Cham: Springer International Publishing, 2015, pp. 563–571. doi: 10.1007/978-3-319-12012-6_62.

[47] A. Sushma and G. S. Veena, “Kannada handwritten word conversion to electronic textual format using HMM model,” in *2016 International conference on computation system and information technology for sustainable solutions (CSITSS)*, IEEE, 2016, pp. 330–335. doi: 10.1109/CSITSS.2016.7779380.

[48] S. Gaur, S. Sonkar, and P. P. Roy, “Generation of synthetic training data for handwritten Indic script recognition,” in *2015 13th International Conference on Document Analysis and Recognition (ICDAR)*, IEEE, 2015, pp. 491–495. doi: 10.1109/ICDAR.2015.7333810.

[49] S. Bhowmik, S. Polley, Md. G. Roushan, S. Malakar, R. Sarkar, and M. Nasipuri, “A holistic word recognition technique for handwritten Bangla words,” *Int. J. Appl. Pattern Recognit.*, vol. 2, no. 2, p. 142, 2015, doi: 10.1504/IJAPR.2015.069539.

[50] S. Thomas, C. Chatelain, L. Heutte, T. Paquet, and Y. Kessentini, “A deep HMM model for multiple keywords spotting in handwritten documents,” *Pattern Anal. Appl.*, vol. 18, no. 4, pp. 1003–1015, Nov. 2015, doi: 10.1007/s10044-014-0433-3.

[51] B. Balci, D. Saadati, and D. Shiferaw, “Handwritten text recognition using deep learning,” *CS231n Convolutional Neural Netw. Vis. Recognit. Stanf. Univ. Course Proj. Rep. Spring*, pp. 752–759, 2017.

[52] K. Dutta, P. Krishnan, M. Mathew, and C. V. Jawahar, “Improving CNN-RNN hybrid networks for handwriting recognition,” in *2018 16th international conference on frontiers in handwriting recognition (ICFHR)*, IEEE, 2018, pp. 80–85. doi: 10.1109/ICFHR-2018.2018.00023.

[53] P. Krishnan and C. V. Jawahar, “Generating synthetic data for text recognition,” *ArXiv Prepr. ArXiv160804224*, 2016, Accessed: May 22, 2025. [Online]. Available: https://arxiv.org/abs/1608.04224

[54] C. Wigington, S. Stewart, B. Davis, B. Barrett, B. Price, and S. Cohen, “Data augmentation for recognition of handwritten words and lines using a CNN-LSTM network,” in *2017 14th IAPR international conference on document analysis and recognition (ICDAR)*, IEEE, 2017, pp. 639–645. doi: 10.1109/ICDAR.2017.110.

[55] B. Wicht, A. Fischer, and J. Hennebert, “Deep learning features for handwritten keyword spotting,” in *2016 23rd International Conference on Pattern Recognition (ICPR)*, IEEE, 2016, pp. 3434–3439. doi: 10.1109/ICPR.2016.7900165.

[56] X. Wu, Q. Chen, J. You, and Y. Xiao, “Unconstrained offline handwritten word recognition by position embedding integrated resnets model,” *IEEE Signal Process. Lett.*, vol. 26, no. 4, pp. 597–601, 2019.

[57] A. M. M. O. Chacko and P. M. Dhanya, “A Comparative Study of Different Feature Extraction Techniques for Offline Malayalam Character Recognition,” in *Computational Intelligence in Data Mining - Volume 2*, vol. 32, L. C. Jain, H. S. Behera, J. K. Mandal, and D. P. Mohapatra, Eds., in Smart Innovation, Systems and Technologies, vol. 32. , New Delhi: Springer India, 2015, pp. 9–18. doi: 10.1007/978-81-322-2208-8_2.

[58] M. Ghosh, S. Malakar, S. Bhowmik, R. Sarkar, and M. Nasipuri, “Feature Selection for Handwritten Word Recognition Using Memetic Algorithm,” in *Advances in Intelligent Computing*, vol. 687, J. K. Mandal, P. Dutta, and S. Mukhopadhyay, Eds., in Studies in Computational Intelligence, vol. 687. , Singapore: Springer Singapore, 2019, pp. 103–124. doi: 10.1007/978-981-10-8974-9_6.

[59] Z. Imani, Z. Ahmadyfard, and A. Zohrevand, “Holistic Farsi handwritten word recognition using gradient features,” *J. AI Data Min.*, vol. 4, no. 1, pp. 19–25, 2016.

[60] S. Malakar, P. Sharma, P. K. Singh, M. Das, R. Sarkar, and M. Nasipuri, “A holistic approach for handwritten Hindi word recognition,” *Int. J. Comput. Vis. Image Process. IJCVIP*, vol. 7, no. 1, pp. 59–78, 2017, doi: 10.4018/IJCVIP.2017010104.

[61] S. Barua, S. Malakar, S. Bhowmik, R. Sarkar, and M. Nasipuri, “Bangla Handwritten City Name Recognition Using Gradient-Based Feature,” in *Proceedings of the 5th International Conference on Frontiers in Intelligent Computing: Theory and Applications*, vol. 515, S. C. Satapathy, V. Bhateja, S. K. Udgata, and P. K. Pattnaik, Eds., in Advances in Intelligent Systems and Computing, vol. 515. , Singapore: Springer Singapore, 2017, pp. 343–352. doi: 10.1007/978-981-10-3153-3_34.

[62] A. Shruthi and M. S. Patel, “Offline handwritten word recognition using multiple features with SVM classifier for holistic approach,” *Int. J. Innov. Res. Comput. Commun. Eng.*, vol. 3, no. 6, pp. 5989–5995, 2015.

[63] N. A. Aouadi and A. K. Echi, “Word extraction and recognition in arabic. handwritten Text,” *Int. J. Comput. Inf. Sci.*, vol. 12, no. 01, 2016, doi: 10.21700/ijcis.2016.103.

[64] S. Das, P. K. Singh, S. Bhowmik, R. Sarkar, and M. Nasipuri, “A harmony search based wrapper feature selection method for holistic bangla word recognition,” *Procedia Comput. Sci.*, vol. 89, pp. 395–403, 2016, doi: 10.1016/j.procs. 2016.06.087.

[65] P. R. Paneri, R. Narang, and M. M. Goswami, “Offline handwritten Gujarati word recognition,” in *2017 Fourth international conference on image information processing (ICIIP)*, IEEE, 2017, pp. 1–5. doi: 10.1109/ICIIP.2017.8313708.

[66] S. Gunter and H. Bunke, “Creation of classifier ensembles for handwritten word recognition using feature selection algorithms,” in *Proceedings Eighth International Workshop on Frontiers in Handwriting Recognition*, IEEE, 2002, pp. 183–188. doi: 10.1109/IWFHR.2002.1030906.

[67] A. Poznanski and L. Wolf, “Cnn-n-gram for handwriting word recognition,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 2305–2314. doi: 10.1109/CVPR.2016.253.

[68] P. Krishnan, K. Dutta, and C. V. Jawahar, “Word spotting and recognition using deep embedding,” in *2018 13th IAPR International Workshop on Document Analysis Systems (DAS)*, IEEE, 2018, pp. 1–6. doi: 10.1109/DAS.2018.70.

[69] K. Dutta, P. Krishnan, M. Mathew, and C. V. Jawahar, “Towards Accurate Handwritten Word Recognition for Hindi and Bangla,” in *Computer Vision, Pattern Recognition, Image Processing, and Graphics*, vol. 841, R. Rameshan, C. Arora, and S. Dutta Roy, Eds., in Communications in Computer and Information Science, vol. 841. , Singapore: Springer Singapore, 2018, pp. 470–480. doi: 10.1007/978-981-13-0020-2_41.

[70] S. Ukil, S. Ghosh, S. M. Obaidullah, K. C. Santosh, K. Roy, and N. Das, “Deep Learning for Word-Level Handwritten Indic Script Identification,” in *Recent Trends in Image Processing and Pattern Recognition*, vol. 1380, K. C. Santosh and B. Gawali, Eds., in Communications in Computer and Information Science, vol. 1380. , Singapore: Springer Singapore, 2021, pp. 499–510. doi: 10.1007/978-981-16-0507-9_42.

[71] P. J. Jino, K. Balakrishnan, and U. Bhattacharya, “Offline Handwritten Malayalam Word Recognition Using a Deep Architecture,” in *Soft Computing for Problem Solving*, vol. 816, J. C. Bansal, K. N. Das, A. Nagar, K. Deep, and A. K. Ojha, Eds., in Advances in Intelligent Systems and Computing, vol. 816. , Singapore: Springer Singapore, 2019, pp. 913–925. doi: 10.1007/978-981-13-1592-3_73.

[72] C. Adak, B. B. Chaudhuri, and M. Blumenstein, “Offline cursive Bengali word recognition using CNNs with a recurrent model,” in *2016 15th International conference on frontiers in handwriting recognition (ICFHR)*, IEEE, 2016, pp. 429–434. doi: 10.1109/ICFHR.2016.0086.

[73] J. Das Gupta, S. Samanta, and B. Chanda, “Ensemble classifier-based off-line handwritten word recognition system in holistic approach,” *IET Image Process.*, vol. 12, no. 8, pp. 1467–1474, Aug. 2018, doi: 10.1049/iet-ipr.2017.0745.

[74] M. Elleuch, N. Tagougui, and M. Kherallah, “Deep Learning for Feature Extraction of Arabic Handwritten Script,” in *Computer Analysis of Images and Patterns*, vol. 9257, G. Azzopardi and N. Petkov, Eds., in Lecture Notes in Computer Science, vol. 9257. , Cham: Springer International Publishing, 2015, pp. 371–382. doi: 10.1007/978-3-319-23117-4_32.

[75] M. Amrouch and M. Rabi, “Deep Neural Networks Features for Arabic Handwriting Recognition,” in *Advanced Information Technology, Services and Systems*, vol. 25, M. Ezziyyani, M. Bahaj, and F. Khoukhi, Eds., in Lecture Notes in Networks and Systems, vol. 25. , Cham: Springer International Publishing, 2018, pp. 138–149. doi: 10.1007/978-3-319-69137-4_14.

[76] C. Oprean, L. Likforman-Sulem, A. Popescu, and C. Mokbel, “Handwritten word recognition using Web resources and recurrent neural networks,” *Int. J. Doc. Anal. Recognit. IJDAR*, vol. 18, pp. 287–301, 2015.

[77] L. Kang, J. I. Toledo, P. Riba, M. Villegas, A. Fornés, and M. Rusiñol, "Convolve, Attend and Spell: An Attention-based Sequence-to-Sequence Model for Handwritten Word Recognition," in *Pattern Recognition*, vol. 11269, T. Brox, A. Bruhn, and M. Fritz, Eds., in Lecture Notes in Computer Science, vol. 11269. , Cham: Springer International Publishing, 2019, pp. 459–472. doi: 10.1007/978-3-030-12939-2_32.

[78] N. Ghanmi, A.-M. Awal, and N. Kooli, "Dynamic Bayesian networks for handwritten Arabic word recognition," in *2017 1st International Workshop on Arabic Script Analysis and Recognition (ASAR)*, IEEE, 2017, pp. 104–108. doi: 10.1109/ASAR.2017.8067769.

[79] G. Chen, Y. Li, and S. N. Srihari, "Word Recognition with Deep Conditional Random Fields," Dec. 04, 2016, *arXiv*: arXiv:1612.01072. doi: 10.48550/arXiv.1612.01072.

[80] S. Kumar, "A study for handwritten Devanagari word recognition," in *2016 International Conference on Communication and Signal Processing (ICCSP)*, IEEE, 2016, pp. 1009–1014. doi: 10.1109/ICCSP.2016.7754301.

[81] P. Krishnan, K. Dutta, and C. V. Jawahar, "Deep feature embedding for accurate recognition and retrieval of handwritten text," in *2016 15th International Conference on Frontiers in Handwriting Recognition (ICFHR)*, IEEE, 2016, pp. 289–294. doi: 10.1109/ICFHR.2016.0062.

[82] B. Stuner, C. Chatelain, and T. Paquet, "Cascading BLSTM networks for handwritten word recognition," in *2016 23rd International Conference on Pattern Recognition (ICPR)*, IEEE, 2016, pp. 3416–3421. doi: 10.1109/ICPR.2016.7900162.

[83] M. Elleuch, N. Tagougui, and M. Kherallah, "Towards Unsupervised Learning for Arabic Handwritten Recognition Using Deep Architectures," in *Neural Information Processing*, vol. 9489, S. Arik, T. Huang, W. K. Lai, and Q. Liu, Eds., in Lecture Notes in Computer Science, vol. 9489. , Cham: Springer International Publishing, 2015, pp. 363–372. doi: 10.1007/978-3-319-26532-2_40.

[84] P. P. Roy, G. Zhong, and M. Cheriet, "Tandem hidden Markov models using deep belief networks for offline handwriting recognition," *Front. Inf. Technol. Electron. Eng.*, vol. 18, no. 7, pp. 978–988, 2017, doi: 10.1631/FITEE.1600996.

[85] R. Almodfer, S. Xiong, M. Mudhsh, and P. Duan, "Multi-column Deep Neural Network for Offline Arabic Handwriting Recognition," in *Artificial Neural Networks and Machine Learning – ICANN 2017*, vol. 10614, A. Lintas, S. Rovetta, P. F. M. J. Verschure, and A. E. P. Villa, Eds., in Lecture Notes in Computer Science, vol. 10614. , Cham: Springer International Publishing, 2017, pp. 260–267. doi: 10.1007/978-3-319-68612-7_30.

[86] M. Amrouch, M. Rabi, and Y. Es-Saady, "Convolutional Feature Learning and CNN Based HMM for Arabic Handwriting Recognition," in *Image and Signal Processing*, vol. 10884, A. Mansouri, A. El Moataz, F. Nouboud, and D. Mammass, Eds., in Lecture Notes in Computer Science, vol. 10884. , Cham: Springer International Publishing, 2018, pp. 265–274. doi: 10.1007/978-3-319-94211-7_29.

[87] D. Das, D. R. Nayak, R. Dash, B. Majhi, and Y. Zhang, "H-WordNet: a holistic convolutional neural network approach for handwritten word recognition," *IET*

Image Process., vol. 14, no. 9, pp. 1794–1805, Jul. 2020, doi: 10.1049/iet-ipr.2019.1398.

[88] S. Malakar, S. Paul, S. Kundu, S. Bhowmik, R. Sarkar, and M. Nasipuri, "Handwritten word recognition using lottery ticket hypothesis based pruned CNN model: a new benchmark on CMATERdb2.1.2," *Neural Comput. Appl.*, vol. 32, no. 18, pp. 15209–15220, Sep. 2020, doi: 10.1007/s00521-020-04872-0.

[89] T. M. Rath and R. Manmatha, "Word spotting for historical documents," *Int. J. Doc. Anal. Recognit. IJDAR*, vol. 9, no. 2–4, pp. 139–152, Apr. 2007, doi: 10.1007/s10032-006-0027-8.

[90] A. Fischer, A. Keller, V. Frinken, and H. Bunke, "Lexicon-free handwritten word spotting using character HMMs," *Pattern Recognit. Lett.*, vol. 33, no. 7, pp. 934–942, 2012.

[91] V. Rowtula and S. R. Oota, "Towards automated evaluation of handwritten assessments," in *2019 International Conference on Document Analysis and Recognition (ICDAR)*, IEEE, 2019, pp. 426–433. doi: 10.1109/ICDAR.2019.00075.

[92] O. Tüselmann and G. A. Fink, "Neural models for semantic analysis of handwritten document images," *Int. J. Doc. Anal. Recognit. IJDAR*, vol. 27, no. 3, pp. 245–263, Sep. 2024, doi: 10.1007/s10032-024-00477-8.

[93] A. Pal, A. Mondal, and C. V. Jawahar, "Advancing Question Answering on Handwritten Documents: A State-of-the-Art Recognition-Based Model for HW-SQuAD," Jul. 15, 2024, *arXiv*: arXiv:2406.17437. doi: 10.48550/arXiv.2406.17437.

[94] "Trier, Ø. D., & Jain, A. K. (1995). Goal-Directed... - Google Scholar." Accessed: Jul. 01, 2025. [Online]. Available: https://scholar.google.com/scholar?hl=en&as_sdt=0%2C5&q=Trier%2C+%C3%98.+D.%2C+%26+Jain%2C+A.+K.+%281995%29.+Goal-Directed+Evaluation+of+Binarization+Methods.+IEEE+Transactions+on+Pattern+Analysis+and+Machine+Intelligence%2C+17%2812%29%2C+1191-1201.&btnG=

[95] "Zuiderveld, K. (1994). Contrast Limited Adaptive... - Google Scholar." Accessed: Jul. 01, 2025. [Online]. Available: https://scholar.google.com/scholar?hl=en&as_sdt=0%2C5&q=Zuiderveld%2C+K.+%281994%29.+Contrast+Limited+Adaptive+Histogram+Equalization.+In+P.+S.+Heckbert+&btnG=

[96] E. A. Sekehravani, E. Babulak, and M. Masoodi, "Implementing canny edge detection algorithm for noisy image," *Bull. Electr. Eng. Inform.*, vol. 9, no. 4, pp. 1404–1410, 2020.

[97] J. Ding, Z. Lin, and L. Yu, "A correction algorithm for document images based on edge contour," in *2015 International Conference on Industrial Technology and Management Science*, Atlantis Press, 2015, pp. 105–108. doi: 10.2991/itms-15.2015.27.

[98] J. Canny, "A computational approach to edge detection," *IEEE Trans. Pattern Anal. Mach. Intell.*, no. 6, pp. 679–698, 1986, doi: 10.1109/TPAMI.1986.4767851.

[99] B. Biswas, U. Bhattacharya, and B. B. Chaudhuri, “Document image skew detection and correction: A survey,” 2023, Accessed: Jun. 27, 2025. [Online]. Available: https://www.academia.edu/download/107101235/IJIRT158871_PAPER.pdf

[100] J. Matas, C. Galambos, and J. Kittler, “Robust detection of lines using the progressive probabilistic hough transform,” *Comput. Vis. Image Underst.*, vol. 78, no. 1, pp. 119–137, 2000.

[101] P. V. Hough, “Method and means for recognizing complex patterns,” Dec. 18, 1962 Accessed: Jun. 27, 2025. [Online]. Available: https://patents.google.com/patent/US3069654/en

[102] N. Otsu, “A threshold selection method from gray-level histograms,” *Automatica*, vol. 11, no. 285–296, pp. 23–27, 1975.

[103] A. Mordvintsev and K. Abid, “Opencv-python tutorials documentation,” *Obtenido Httpsmedia Readthedocs Orgpdfopencv-Python-Tutroalslatestopencv-Python-Tutroals Pdf*, 2014, Accessed: Jul. 02, 2025. [Online]. Available: https://www.kaggle.com/blobs/download/forum-message-attachment-files/16192/OpenCV-Python%20Tutorials-2017.pdf

[104] R. C. Gonzalez, *Digital image processing*. Pearson education india, 2009. Accessed: Jul. 03, 2025. [Online]. Available: https://books.google.com/books?hl=en&lr=&id=a62xQ2r_f8wC&oi=fnd&pg=PA19&dq=Gonzalez,+R.+C.,+%26+Woods,+R.+E.+(2018).+Digital+Image+Processing+(4th+ed.).+Pearson.&ots=3B4tP1pIYD&sig=fKV7YxH1jBdCDlPtCkYXtnuy8J4

[105] S. Suzuki, “Topological structural analysis of digitized binary images by border following,” *Comput. Vis. Graph. Image Process.*, vol. 30, no. 1, pp. 32–46, 1985.

[106] “facebook/deit-base-patch16-224 · Hugging Face.” Accessed: Jul. 30, 2025. [Online]. Available: https://huggingface.co/facebook/deit-base-patch16-224

[107] H. Touvron, M. Cord, M. Douze, F. Massa, A. Sablayrolles, and H. Jégou, “Training data-efficient image transformers & distillation through attention,” in *International conference on machine learning*, PMLR, 2021, pp. 10347–10357. Accessed: Jul. 29, 2025. [Online]. Available: https://proceedings.mlr.press/v139/touvron21a

[108] A. Dosovitskiy *et al.*, “An image is worth 16x16 words: Transformers for image recognition at scale,” *ArXiv Prepr. ArXiv201011929*, 2020, Accessed: Jul. 31, 2025. [Online]. Available: https://arxiv.org/pdf/2010.11929/1000

[109] A. Vaswani *et al.*, “Attention is all you need,” *Adv. Neural Inf. Process. Syst.*, vol. 30, 2017, Accessed: Jul. 31, 2025. [Online]. Available: https://proceedings.neurips.cc/paper/2017/hash/3f5ee243547dee91fbd053c1c4a845aa-Abstract.html

[110] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778. doi: 10.1109/CVPR.2016.90.

[111] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, “Bert: Pre-training of deep bidirectional transformers for language understanding,” in *Proceedings of the 2019 conference of the North American chapter of the association for computational linguistics: human language technologies, volume 1 (long and short papers)*, 2019, pp. 4171–4186. doi: 10.18653/v1/N19-1423.

[112] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, “Densely connected convolutional networks,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 4700–4708. doi: https://doi.org/10.48550/arXiv.1608.06993.

[113] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, “Mobilenetv2: Inverted residuals and linear bottlenecks,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 4510–4520. doi: https://doi.org/10.48550/arXiv.1801.04381.

[114] K. Simonyan and A. Zisserman, “Very Deep Convolutional Networks for Large-Scale Image Recognition,” Apr. 10, 2015, *arXiv*: arXiv:1409.1556. doi: 10.48550/arXiv.1409.1556.

[115] A. Graves, S. Fernández, and J. Schmidhuber, “Bidirectional LSTM Networks for Improved Phoneme Classification and Recognition,” in *Artificial Neural Networks: Formal Models and Their Applications – ICANN 2005*, vol. 3697, W. Duch, J. Kacprzyk, E. Oja, and S. Zadrożny, Eds., in Lecture Notes in Computer Science, vol. 3697. , Berlin, Heidelberg: Springer Berlin Heidelberg, 2005, pp. 799–804. doi: 10.1007/11550907_126.

[116] M. Tan and Q. Le, “Efficientnet: Rethinking model scaling for convolutional neural networks,” in *International conference on machine learning*, PMLR, 2019, pp. 6105–6114. Accessed: Sep. 16, 2025. [Online]. Available: https://proceedings.mlr.press/v97/tan19a.html?ref=ji

[117] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra, “Grad-cam: Visual explanations from deep networks via gradient-based localization,” in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 618–626. Accessed: Sep. 26, 2025. [Online]. Available: http://openaccess.thecvf.com/content_iccv_2017/html/Selvaraju_Grad-CAM_Visual_Explanations_ICCV_2017_paper.html

[118] D. M. Blei, A. Y. Ng, and M. I. Jordan, “Latent dirichlet allocation,” *J. Mach. Learn. Res.*, vol. 3, no. Jan, pp. 993–1022, 2003.

[119] C. D. Manning, *Introduction to information retrieval*. Syngress Publishing, 2008. Accessed: Sep. 29, 2025. [Online]. Available: http://diglib.globalcollege.edu.et:8080/xmlui/bitstream/handle/123456789/1096/Manning_introduction_to_information_retrieval.pdf?sequence=1&isAllowed=y

[120] R. Řeh\uuřek and P. Sojka, “Software framework for topic modelling with large corpora,” 2010, Accessed: Sep. 30, 2025. [Online]. Available: https://repozitar.cz/publication/15725/?lang=cs;kod=S530

[121] G. Salton and C. Buckley, “Term-weighting approaches in automatic text retrieval,” *Inf. Process. Manag.*, vol. 24, no. 5, pp. 513–523, 1988.

[122] A. Singhal, “Modern information retrieval: A brief overview,” *IEEE Data Eng Bull*, vol. 24, no. 4, pp. 35–43, 2001.

[123] “CMATERdb2.1.2: A Handwritten Bangla word Database.” [Online]. Available: https://www.kaggle.com/datasets/showmikbhowmik/cmaterdb212-a-handwritten-bangla-word-database

LIST OF PUBLICATIONS

- Anju, A. T., Binu P. Chacko, and KP Mohamed Basheer. "Review of offline handwritten text recognition in south Indian languages." Malaya Journal of Matematik (MJM) 9.1, 2021 (2021): 751-756, https://doi.org/10.26637/MJM0901/0132.

- Anju AT, ChackoBP, Basheer KPM. (2024) Advancements in Segmentation of Unconstrained Malayalam Handwritten Documents for Enhanced OCR. Indian Journal of Science and Technology. 17(36):3763-3775. https://doi.org/10.17485/IJST/v17i36.2220

- A paper titled "Vision Transformer and Hybrid Models for Malayalam Handwritten Word Recognition" is accepted in International journal of artificial intelligence (June 2026 issue).

CONFERENCE PROCEEDINGS

- Anju, A. T., Chacko, B. P., & KP, M. B. (2022, December). Segmentation-free Offline Handwritten Malayalam Word Recognition using Transfer Learning Based Deep Neural Network. In 2022 International Conference on Knowledge Engineering and Communication Systems (ICKES) (pp. 1-6). IEEE, DOI: 10.1109/ICKECS56523.2022.10060557.

- Anju, A. T., Chacko, B. P., & Mohamed Basheer, K. P. (2022, November). HOG Feature-Based Offline Handwritten Malayalam Word Clustering with Lexicon Reduction. In International Conference on Innovations in Computational Intelligence and Computer Vision (pp. 607-617). Singapore: Springer Nature Singapore, DOI: 10.1007/978-981-99-2602-2_46.

- Anju, A. T., Chacko, B. P., & Basheer, K. P. (2024, July). Text line segmentation of offline malayalam handwritten document. In AIP Conference Proceedings (Vol. 2965, No. 1). AIP Publishing, SCOPUS, DOI:10.1063/5.0211999.

- Anju, A. T., Chacko, B. P., & KP, M. B. (2024, December). TrA-MHWR: Transformer-Enhanced Feature Extraction with Attention-Based Recognition for Offline Handwritten Malayalam Words. In 2024 International Conference on Emerging Research in Computational Science (ICERCS) (pp. 1-7). IEEE. DOI: 10.1109/ICERCS63125.2024.10894892.