# FDRE TECHNICAL AND VOCATIONAL TRAINING INSTITUTE

**(FDRE TVTI)**

**DEPARTMENT OF INFORMATION TECHNOLOGY**



የኢፌዲሪ የቴክኒክና ሙያ ስልጠና ኢንስቲትዩት
FDRE TECHNICAL & VOCATIONAL TRAINING INSTITUTE

**DISSERTATION TITLE:**

# AMHARIC SCANNED DOCUMENT RETRIEVAL WITH APPROACH OF NLP

**A DISSERTATION SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF**

**MASTER OF SCIENCE IN INFORMATION TECHNOLOGY**


**BY**

**CHERNET TAYE DARE**

Student ID: TTMR071/16

**Principal Advisor: Birkinesh W/Yohanis (phd)**


**Addis Ababa, Ethiopia**

**June 2026**

**FDRE TECHNICAL AND VOCATIONAL TRAINING INSTITUTE**

**COLLEGE OF GRADUATE STUDIES**

**DEPARTMENT OF INFORMATION TECHNOLOGY**

# AMHARIC SCANNED DOCUMENT RETRIEVAL WITH APPROACH OF NLP

**By: Chernet Taye Dare**

Student ID: TTMR071/16

*A dissertation submitted to the Department of Information Technology, College of Graduate Studies, FDRE Technical and Vocational Training Institute (FDRE TVTI), in partial fulfillment of the requirements for the degree of Master of Science in Information Technology.*

**Principal Advisor: Birkinesh W/Yohanis (phd)**

**Addis Ababa, Ethiopia**

**June 2026**

# DECLARATION

I, the undersigned, declare that this dissertation entitled "Amharic Scanned Document Retrieval with Approach of NLP" is my own original work, prepared under the supervision of my advisor(s). It has not been submitted for the award of any degree, diploma, or any other academic qualification in this or any other institution. All sources of materials used in this dissertation have been duly acknowledged.

**Declared by:**

Name: Chernet Taye

Signature: _____

Department: Information Technology, FDRE TVTI

Date: _____

This dissertation has been submitted for examination with my approval as the candidate's advisor.

Principal Advisor: Birkinesh W/Yohanis (phd)

Signature: 

Date: _____

# APPROVAL PAGE

This is to certify that the dissertation entitled "Amharic Scanned Document Retrieval with Approach of NLP" submitted by Chernet Taye Dare ID No. TTMR 071/2016, in partial fulfilment of the requirements for the degree of Master of Science in Information Technology, has been examined and approved by the Board of Examiners of the Department of Information Technology, FDRE Technical and Vocational Training Institute.

| Examination Role | Name | Signature | Date |
|---|---|---|---|
| Principal Advisor | Birkinesh W/Yohanis (phd) |  | |
| Co-Advisor (if any) | _____ | _____ | _____ |
| Internal Examiner | _____ | _____ | _____ |
| External Examiner | _____ | _____ | _____ |
| Department Head, IT | _____ | _____ | _____ |
| Dean, College of Graduate Studies | _____ | _____ | _____ |

Place: Addis Ababa, Ethiopia

Date of Final Approval: _____

# ACKNOWLEDGEMENTS

First and foremost, I would like to express my sincere gratitude to Almighty God for granting me the strength and patience to complete this dissertation.

I am deeply indebted to my Principal Advisor, Birkinesh W/Yohanis (phd) for the invaluable guidance, constructive criticism, and continuous encouragement throughout this study for her support and insightful comments.

My sincere thanks to the Department of Information Technology and the College of Graduate Studies of FDRE TVTI for providing the academic environment, laboratory facilities, and resources required to complete this research.

Finally, I owe a deep debt of gratitude to my family, friends, and classmates for their unwavering support, patience, and encouragement throughout my postgraduate studies.

# ABSTRACT

Ethiopia possesses a vast and valuable collection of documents written in Amharic, ranging from administrative records and legal documents to historical and archival materials. In recent years, many of these documents have been digitized through scanning in an effort to preserve them and improve accessibility. However, despite digitization, effective access to the information contained in these scanned documents remains a major challenge.The primary difficulty arises from the limitations of existing Optical Character Recognition (OCR) technologies when applied to the Amharic script. Due to the visual complexity of the Ethiopic writing system, the morphological richness of the language, and variations in document quality, OCR systems often generate noisy and error-prone text. As a result, conventional information retrieval systems that rely on exact keyword matching perform poorly, making it difficult for users to locate relevant documents efficiently.This study addresses these challenges by proposing a rule-based Amharic scanned document retrieval system using Natural Language Processing techniques. The research follows a design science research approach, involving data acquisition from scanned Amharic documents, identification and formalization of linguistic and OCR-related rules, system design and implementation, and systematic evaluation using standard information retrieval metrics. The rulse-based approach enables the explicit handling of common OCR errors, Amharic morphological variations, and character ambiguities that negatively affect retrieval performance.The expected outcomes of this research include the development of a functional prototype that demonstrates improved retrieval accuracy and efficiency, enhanced understanding of Amharic language processing challenges in scanned document environments, and a meaningful contribution to the growing field of Amharic Natural Language Processing. Ultimately, the study aims to support Ethiopian institutions by improving access to digitized Amharic documents and strengthening digital document management practices.

# Table of Contents

CHAPTER ONE ..... 1
INTRODUCTION ..... 1
1.1 Background of the Study ..... 1
1.2 Statement of the Problem ..... 2
1.3 Objectives of the Study ..... 3
1.3.1 General Objective ..... 3
1.3.2 Specific Objectives ..... 3
1.4 Research Questions ..... 3
1.5 Hypothesis of the Study ..... 4
1.6 Scope / Delimitations of the Study ..... 4
1.7 Significance of the Study ..... 5
1.8 Limitations of the Study ..... 6
1.9 Organization of the Paper ..... 7
CHAPTER TWO ..... 8
Literature Review ..... 8
2.1 Theoretical Review of Amharic and Natural Language Processing ..... 8
2.2 Amharic Language ..... 9
2.2.1 The Amharic Writing System ..... 9
2.2.2 Problems in Amharic Spelling ..... 10
2.2.3 Amharic Script's nature ..... 11
2.2.4 Grammar of the Amharic Language ..... 11
2.2.5 Amharic Morphology ..... 14
2.2.6 Amharic Attachments ..... 14
2.2.7 Compound Words in Amharic ..... 14
2.2.8 Punctuation in Amharic ..... 15
2.2.9 Numerical digits in Amharic ..... 15
2.3 Natural Language Processing ..... 15
2.3.1 Morphological Synthesis ..... 15
2.3.2 Morphological Analysis ..... 16
2.3.3 Syntactic Analysis ..... 16

2.3.4 Semantic Analysis ..... 17
2.3.5 Discourse Integration ..... 17
2.3.6 Pragmatic Analysis ..... 17
2.3.7 Lexical Analysis ..... 18
2.4 Empirical Literature Review ..... 18
2.4.1 Information Retrieval from Scanned Documents ..... 18
2.4.2 Applications of Amharic NLP ..... 19
2.4.3 NLP Systems in Practical ..... 19
2.5 Conceptual Framework ..... 20
CHAPTER THREE ..... 22
THEORETICAL BACKGROUND OF DOCUMENT RETRIEVAL ..... 22
3.1 Information Retrieval ..... 22
3.1.1 Basic Language Problems in Information Retrieval ..... 23
3.1.2 Query ..... 24
3.1.3 Natural Language Processing in Information Retrieval ..... 25
3.1.4 Document Indexing ..... 27
3.1.5 Term Weighting ..... 32
3.1.6 Ranking ..... 34
3.1.7 Documents Database ..... 35
3.1.8 Information Retrieval Models ..... 36
3.1.9 Evaluation Techniques for IR Systems ..... 39
3.2 Document Image Retrieval ..... 41
3.2.1 Document Image Processing Stages ..... 43
3.2.2 Feature Extraction in OCR ..... 46
3.2.4 Similarity Measures ..... 47
3.3 Related Works ..... 50
3.4 English and Other Non-Amharic Document Images Retrieval ..... 51
3.4.1 Amharic Document Images Retrieval ..... 52
CHAPTER FOUR ..... 54
PROPOSED SYSTEM ARCHITECTURE ..... 54
4.1 proposed system and experimental results for Amharic document retrieval ..... 54

4.2 Amharic Word Synthesizer and Textual Feature Corpus Preparation ..... 57
4.3 Document Image processing ..... 59
4.3.1 Noise Removal and experiment ..... 62
4.3.2 Binarization and experiment ..... 63
4.3.3 Line segmentation and experiment ..... 65
4.3.4 Word Segmentation and experiment ..... 67
4.4 NLP Processing ..... 69
4.5 Document Indexing ..... 70
4.6 Query Processor ..... 71
4.6.1 Searching Techniques ..... 75
4..6.2 Matching experiment ..... 77
4.7 Ranking search result ..... 80
4.8 Contribution of Research ..... 81
4.10 Experimental Resources ..... 81
4.10.1 Data Sets used ..... 82
4.11 Test Result ..... 83
CHAPTER FIVE ..... 88
CONCLUSIONS AND RECOMMENDATIONS ..... 88
5.1 Conclusions ..... 88
5.2 Recommendations ..... 89
REFERENCES ..... 90
APPENDIX I: The Amharic Character and their Roman counter parts ..... 95
Appendix II: sample data from different sources ..... 98

# LIST OF TABLES

Table 4.1Result Summary table ..... 86

# LIST OF FIGURES

Figure 2.1 Conceptual Framework of the Study ..... 20
Figure 4.1 Proposed System Architecture .....56
Figure 4.2 document image processing.....61
Figure 4.3 Query Processing flow diagram.....73
Figure 4.4 home page user interface……..…83
Figure 4.5 user interface shows uploading scanned document…….….84
Figure 4.6 user interface shows searching and result……..…….85
Figure 4.7 user interface shows overall retrieval performance and result…… 86

# LIST OF ABBREVIATIONS AND ACRONYMS

| Abbreviation | Full Form |
|---|---|
| NLP | Natural Language Processing |
| ADIRS | Amharic Document Image Retrieval System ADIR |
| IR | Information Retrieval |
| CV | Consonant vowel |
| DBMS | Database Management System DR Document Retrieval |
| ID | Document Identification |
| DIR | Document Image Retrieval |
| DTW | Dynamic time warping |
| IDF | Inverse Document Frequency |
| IDF | Inverse Document Frequency |
| NCC | Normalized Cross-correlation |
| OCR | Optical Character Recognition |
| TF | Term Frequency |
| VSM | Vector Space Model |

# CHAPTER ONE

# INTRODUCTION

## 1.1 Background of the Study

Information Retrieval (IR) is a key area of computer science that focuses on helping people find the information they need from large collections of data. In simple terms, IR systems are designed to take a user's query and return the most relevant documents or results from a vast and often unstructured information space. This has become increasingly important as digital information continues to grow at a very fast rate, making it difficult for users to manually locate useful content.

In practice, IR systems are widely used in everyday technologies such as web search engines, digital libraries, question-answering systems, and enterprise search platforms. These systems help users quickly access relevant knowledge in areas like education, business, law, and research, where large volumes of data must be processed efficiently (Hersh, 2023).

Early IR systems mainly relied on keyword matching techniques, where documents were retrieved based on the presence and frequency of query terms. Classical models such as the vector space model and probabilistic ranking methods like BM25 became widely used because of their simplicity and effectiveness. However, these traditional approaches often struggle to understand the actual meaning behind words, especially when different terms are used to express the same idea (Manning et al., 2018; Mothe, 2022).

In the Ethiopian context, a substantial portion of valuable information—such as historical manuscripts, traditional literature, government decrees, and administrative records—continues to exist primarily in physical, paper-based formats. These documents represent not only administrative resources but also important elements of the nation's cultural and historical heritage. To preserve these materials and improve public access, digitization initiatives have increasingly been undertaken by scanning physical documents and converting them into digital form.

Existing Amharic Optical Character Recognition (OCR) models have shown gradual improvements, they still exhibit notable error rates, particularly when processing documents with mixed font styles, degraded scan quality, or historical and handwritten scripts. In such conditions,

OCR systems often produce recognition errors such as incorrect characters, missing or extra characters, and improper splitting or merging of words. These inaccuracies can significantly affect downstream text processing tasks. In particular, keyword-based search and traditional information retrieval systems become less effective when OCR output contains noise, as incorrect or incomplete text reduces the system's ability to correctly match queries with relevant documents, leading to reduced retrieval accuracy. This challenge has been widely recognized in OCR and document analysis research, where it is noted that recognition errors directly impact retrieval performance (Wende, 2015; Smith, 2016).

Building reliable information retrieval systems for scanned Amharic documents is still a significant challenge in today's digital information world. In general, information retrieval systems are designed to help users find and rank relevant documents from large collections of unstructured data, often using methods like keyword matching and statistical ranking. these traditional approaches tend to struggle when the input text is noisy or imperfect, which is very common in scanned documents and can reduce the accuracy of search results. Because of these limitations, recent research has focused on improving how retrieval systems understand both queries and documents by using more advanced methods that capture meaning more effectively. As a result, modern information retrieval systems are gradually moving toward more intelligent and context-aware approaches that provide better search quality and user satisfaction in complex environments (Manning et al., 2018; Hersh, 2023).

## 1.2 Statement of the Problem

To improve access to information stored in scanned Amharic documents, there is a need for a system that can effectively process and retrieve text from image-based files. Many institutions in Ethiopia, such as universities, libraries, government offices, and research centers, store important documents in scanned formats including books, reports, letters, and historical records. Although these documents are digitally preserved, searching and retrieving information from them remains difficult.

To address the challenges related to Optical Character Recognition (OCR), it is important to reduce the recognition errors that commonly occur in scanned Amharic documents. Because of the complexity of the Ethiopic script, poor image quality, noise, skewed pages, and font variations,

OCR systems often produce incorrect text. These errors reduce the quality of extracted information and negatively affect search accuracy.

To handle the linguistic complexity of the Amharic language, there is also a need for Natural Language Processing (NLP) techniques that can normalize text and reduce word variations. Since Amharic words can appear in different forms depending on grammar and context, traditional keyword-based searching often fails to retrieve all relevant documents accurately.

To overcome these challenges and reduce the limitations of manual searching, this study focuses on developing an Amharic Scanned Document Retrieval System using NLP techniques. The system integrates OCR, text preprocessing, indexing, and similarity-based ranking methods to improve the accuracy, efficiency, and practicality of retrieving information from scanned Amharic documents.

## 1.3 Objectives of the Study

### 1.3.1 General Objective

The general objective of this study is to design, develop, and evaluate a robust Amharic scanned document retrieval system, integrated with Natural Language Processing (NLP) techniques, for effective and accurate information retrieval from scanned Amharic documents.

### 1.3.2 Specific Objectives

To achieve the general objective, the following specific objectives will be pursued:

1. To identify and study common OCR errors that occur in scanned Amharic documents.
2. To develop NLP-based methods that help clean, normalize, and prepare Amharic text for searching.
3. To design and implement a document retrieval system that uses NLP techniques such as tokenization, normalization, stemming, and indexing.
4. To test and evaluate the system using standard performance measures such as Precision, Recall, F-score, and MAP, and compare it with a basic baseline system.

## 1.4 Research Questions

The following research questions will guide the investigation and align directly with the specific objectives of the study:

- What types of OCR errors commonly occur in scanned Amharic documents, and how do these errors affect the performance of existing information retrieval systems?
- What linguistic and structural processing techniques can be applied to better handle the characteristics of the Amharic language and reduce the impact of OCR-related errors?
- How can a Natural Language Processing (NLP)-based system be designed to effectively retrieve information from scanned Amharic documents?
- How well does the proposed system perform in terms of retrieval effectiveness, particularly in measures such as precision and recall, compared to baseline methods?

## 1.5 Hypothesis of the Study

Based on the research problem and objectives, this study hypothesizes that integrating Natural Language Processing (NLP) techniques with OCR processing can significantly improve the accuracy and efficiency of information retrieval from scanned Amharic documents compared to systems that rely only on raw OCR output or basic keyword matching.

It is further assumed that applying NLP-based text processing methods tailored to the characteristics of the Amharic language will help reduce the impact of OCR errors and linguistic variations. This will enable the system to better understand and normalize the extracted text, leading to improved retrieval performance in terms of precision, recall, and overall relevance of search results.

## 1.6 Scope / Delimitations of the Study

This study focuses on developing an Amharic scanned document retrieval system using Natural Language Processing (NLP) techniques. The main goal is to improve how users can search and retrieve information from scanned Amharic documents that are stored in image-based formats.

The study is limited to printed Amharic documents that have been scanned into digital form. These include materials such as books, reports, letters, administrative documents, and other similar text-based records. Handwritten documents are not included in this study because they require more advanced recognition techniques that are beyond the scope of this work.

The system covers the full process of document retrieval, starting from scanned image input, converting images to text using Optical Character Recognition (OCR), processing the extracted text using NLP techniques, and finally retrieving relevant documents using indexing and ranking

methods. The NLP part mainly focuses on cleaning the text, normalizing word forms, breaking text into tokens, and reducing word variations so that search results become more accurate.

The system is designed to support improved keyword-based searching, where user queries are processed and matched against indexed documents in a more effective way. The performance of the system is evaluated using standard information retrieval measures such as precision, recall, F-score, and Mean Average Precision (MAP).

The study has some clear limitations. The performance of the system depends heavily on the quality of the OCR output. If the scanned documents are very low quality, heavily degraded, or unclear, some errors may still remain in the extracted text. In addition, the study uses a limited dataset of scanned Amharic documents, so it may not cover all possible document types, writing styles, or real-world variations.

Another limitation is that the system mainly relies on traditional NLP techniques and does not include advanced deep learning or semantic-based models. Also, while the system is designed to be efficient, large-scale deployment on very big document collections is not the main focus of this study.

Even with these limitations, the system provides a practical and effective approach to improving the retrieval of scanned Amharic documents and can serve as a strong foundation for future improvements and research.

## 1.7 Significance of the Study

This research holds significant practical, academic, and societal implications:

- Practical Benefits for Organizations: Many governmental, archival, and private organizations in Ethiopia possess vast collections of scanned Amharic documents that are currently underutilized due to difficult access. this research will provide a critical tool for efficient digitization, indexing, and retrieval, leading to improved workflow efficiency, faster decision-making, and better access to historical and administrative information.
- Contribution to Amharic NLP: This study will significantly contribute to the nascent field of Amharic Natural Language Processing, particularly in the challenging domain of noisy text derived from scanned documents (Tonja et al., 2023a). It will provide insights into effective strategies for handling Amharic morphology and character variations in a rule-based context.

- Foundation for Future Development: The implemented system and the identified rule sets can serve as a valuable resource and baseline for developing more advanced or hybrid Amharic document processing applications, including specialized search engines, digital libraries, and knowledge management systems.

## 1.8 Limitations of the Study

- Dependency on OCR Quality: The performance of rule-based post-correction and information retrieval systems largely depends on how accurate the initial OCR output is. If the OCR produces many errors, those mistakes usually carry over into later processing steps, making it difficult for correction rules to fully fix the text and ultimately reducing the quality of retrieval results (Belay et al., 2020).
- Rule Engineering Complexity: Designing and maintaining a comprehensive set of linguistic and structural rules is a time-consuming and iterative process in natural language processing systems. As rule sets expand, ensuring consistency, coverage, and adaptability across diverse linguistic patterns, exceptions, and real-world noise becomes increasingly difficult. This complexity can limit scalability and reduce the ability of systems to generalize effectively in large or evolving datasets, particularly in low-resource language environments (Jurafsky & Martin, 2023).
- Generalizability of Rules: Rules developed for a specific domain or type of Amharic document (e.g., administrative texts) might require significant adaptation or expansion to be fully effective across vastly different domains (e.g., poetic texts, historical manuscripts with archaic script).
- Dataset Representativeness: The evaluation dataset, while carefully curated, may not perfectly capture the full spectrum of variations in scanned Amharic document quality, fonts, and linguistic styles encountered in real-world scenarios.
- Computational Resources: While rule-based systems are generally less computationally intensive than deep learning models for training, large-scale processing of documents may still require adequate computational resources.

## 1.9 Organization of the Paper

This thesis proposal is structured into several chapters to systematically present the research plan:

- ✓ **Chapter 1: Introduction:** Provides an overview of the research area, statements the problem, outlines the objectives, research questions, and hypothesis, defines the scope and delimitations, highlights the significance, and acknowledges the limitations of the study.
- ✓ **Chapter 2: Review of Related Literature:** Presents a comprehensive review of existing theoretical concepts and empirical studies related to information retrieval, Natural Language Processing (with a focus on Amharic NLP) tokenization, stemming/lemmatization, morphological analysis, Optical Character Recognition, and rule-based systems. It also includes a conceptual framework illustrating the core components and relationships in the proposed system.
- ✓ **Chapter 3: Research Methodologies:** Details the research design and pattern, data collection strategies, the proposed system architecture, the methodology for rule identification and formalization, implementation details, and the evaluation framework including metrics and experimental setup.
- ✓ **Chapter 4: System Implementation, Experimentation and Results:** This chapter, in the full thesis, would detail rule-based system architectural design its various modules (e.g., OCR post-correction, tokenizer, morphological analyzer, indexer, query processor),the specific implementation choices and in the full thesis, would present the experimental setup, the results of evaluating the developed system against the defined metrics, and a detailed analysis of these results,
- ✓ **Chapter 5: and summarize:** conclusion of the main findings, discuss the implications of the research, and suggest directions for future research.

# CHAPTER TWO

# Literature Review

This chapter provides a general overview of the theory base and existing empirical work relevant to Amharic scanned document retrieval using an NLP approach. It will help to establish the state of the art, delineate the research gaps, and position the intended study within the broader academic literature.

## 2.1 Theoretical Review of Amharic and Natural Language Processing

Amharic is one of the major languages of Ethiopia and is written using the Ethiopic (Ge'ez-based) script, which is also used by several other Ethiopian languages. Recent research highlights that Amharic is a low-resource and morphologically complex language, which makes its computational processing challenging compared to widely studied languages like English (Yeshambel et al., 2022). Despite these challenges, Amharic has become an important focus in Natural Language Processing (NLP) and Information Retrieval (IR) research due to its large number of speakers and its extensive use in government, education, and digital communication.

One of the main challenges in processing Amharic text is its rich and complex morphology, where a single root word can generate many different surface forms through inflection and derivation. This non-linear structure makes traditional keyword-based retrieval less effective because simple word matching cannot capture all variations of the same meaning. Recent studies show that effective Amharic NLP and IR systems require morphological analysis and advanced text representation techniques such as root-based or embedding-based models to improve retrieval accuracy and language understanding (Yeshambel et al., 2023).

In addition, recent developments in Amharic NLP emphasize the importance of building language resources such as corpora, lexicons, and annotated datasets to support machine learning and deep learning models. Studies have shown that integrating morphological processing with modern representation learning techniques, including transformer-based models, significantly improves performance in tasks such as information retrieval, classification, and text understanding for Amharic and other Ethiopian languages (Tonja et al., 2024).

Overall, current research strongly indicates that effective processing of Amharic text requires a combination of linguistic knowledge and modern NLP techniques, especially in information retrieval systems where morphological variation and resource limitations significantly affect system performance

## 2.2 Amharic Language

### 2.2.1 The Amharic Writing System

The Amharic writing system is based on the Ethiopic (Ge'ez) script, one of the oldest writing systems still actively used today. Unlike alphabetic writing systems such as English, the Ethiopic script is syllabic in nature, where each symbol represents a consonant–vowel combination rather than separate consonants and vowels. In modern Amharic, each base character commonly appears in seven different forms, known as orders, with each form representing a particular vowel sound. These vowel changes are expressed through systematic visual modifications of the base character, making vowel representation an embedded part of the writing system itself. This unique structure makes computational processing of Amharic text more challenging than many alphabetic languages.

Amharic is also considered a morphologically rich and low-resource language. A single root word can generate many surface forms through inflection, derivation, and other morphological processes. In addition, the language contains many visually similar fidel characters, spelling variations, and inconsistent representations of compound words across documents. These characteristics create major challenges for Natural Language Processing (NLP), Optical Character Recognition (OCR), and Information Retrieval (IR) systems because semantically related words may appear in several different forms. As a result, traditional keyword-based retrieval approaches often struggle to effectively identify relevant documents (Gezmu et al., 2021).

Recent studies on Amharic information retrieval and language processing emphasize that the morphological complexity and orthographic variability of Amharic significantly affect retrieval effectiveness and text understanding. Researchers further note that advanced preprocessing techniques such as morphological analysis, normalization, stemming, and contextual text representation are necessary for improving retrieval performance in Amharic document processing systems. These challenges become even more significant in scanned document environments,

where OCR systems may introduce character recognition and segmentation errors due to the visual similarity and structural complexity of Ethiopic characters (Yeshambel et al., 2022)

**Relevance to Amharic Image Retrieval with NLP**

The structural properties of the Amharic writing system directly affect the design of Amharic image retrieval system. Because each fidel encodes both consonant and vowel information within its shape, visually different symbols may share the same morphological root. For this reason, NLP techniques such as morphological analysis, vowel normalization, and character decomposition play a crucial role in grouping related word images.

For example, the root "ጸፍ" (to write) produces many surface forms (ጻፈ፣ ጽፋል፣ ጽፈት), each visually distinct in scanned images. Without NLP processing, a system would treat these as unrelated images. By contrast, Amharic NLP method can normalize these variations, extract their underlying structure, and improve the accuracy of image retrieval.

So, understanding the Amharic writing system is essential for building a retrieval model that effectively integrates NLP rules with image-level pattern recognition, forming the foundation for this research.

### 2.2.2 Problems in Amharic Spelling

Amharic orthography is largely phonetic, meaning that words are generally written in ways that reflect their pronunciation. Also spelling variation is still very common across regions, dialects, writing styles, and individual authors. These inconsistencies create major challenges for Natural Language Processing (NLP), Optical Character Recognition (OCR), and Information Retrieval (IR) systems because words with the same meaning may appear in several different written forms. (Yeshambel et al., 2022). Some of the specific issues include:

Ge'ez loan words that introduce redundant graphemes: for example, Amharic uses two series for "s" (ሠ & ሰ), "ts" (ፀ & ጸ) and "h" (ሀ, ኀ, ሐ) inherited from Ge'ez. This causes spelling variation: e.g., the word "sun" may appear as "ጸሀይ", "ፀሀይ", etc.

Abbreviations/slashes: Short forms like "ም/ር" for "መምህር" (teacher) are frequent. Regional/dialectal differences: For example, southern dialect forms like "ሰማይ" vs "ፀማይ" may appear. Loan-words: Foreign words (e.g., computer) may be written as "ኮምፒውተር", "ኮምፕዩተር", "ኮምፒውተር", each variant representing the same meaning.

These aspects complicate retrieval in scanned document systems because string-based indexing may miss alternate forms. NLP layer can map variants to canonical forms or roots, enabling better retrieval. For example, in the study "Amharic Adhoc Information Retrieval System Based on Morphological Features" the authors observe that Amharic words may take up to four prefixes and five suffixes, and thus morphological normalization is critical.

### 2.2.3 Amharic Script's nature

Amharic script is organized so that for each consonant C, there are ordered syllabic forms: C/ ə (C with default vowel), C/u, C/i, C/a, C/ie, C/o etc. For example, "ሀ" (bə), "ቡ" (bu), "ቢ" (bi), "ባ" (ba), "ቤ" (be), "ብ" (bə) and "ቦ" (bo). Thus, the writing system is more a syllabary than a pure alphabet. This structure also means that scanned text may contain ligatures or variant glyph shapes; OCR and post-processing must handle Unicode normalization, ligature decomposition, and variant representations. For example, in the paper "Amharic Document Representation for Adhoc Retrieval" the authors remark on the 34 base characters each having six modified vowel orders. SciTePress Hence, NLP system needs modules to normalize character orders, decompose syllables, and map to canonical forms before indexing.

### 2.2.4 Grammar of the Amharic Language

Amharic grammar is a highly structured and morphologically rich system that plays a crucial role in Natural Language Processing (NLP), especially for applications such as scanned document retrieval systems. Effective interpretation of Amharic text requires understanding its complex grammatical organization, which enables computational systems to capture meaning even when documents are noisy, scanned, or variably written. In Amharic, words are built through a hierarchical linguistic structure that includes phonemes, morphemes, roots, stems, and fully inflected word forms. Phonemes represent the smallest units of sound, while morphemes carry minimal meaning and combine to form meaningful lexical units. Many Amharic words are derived from triliteral root patterns, particularly in verbs and related nominal forms, where meaning is expanded through systematic morphological processes such as affixation, vowel insertion, reduplication, and stem modification.

For Amharic scanned document retrieval systems, correctly identifying and normalizing these morphological variations is essential. Without proper linguistic processing, different surface forms of the same root may be treated as unrelated terms, reducing retrieval accuracy. Recent NLP

studies on Ethiopian languages emphasize that morphological analysis and normalization significantly improve information retrieval and machine understanding of low-resource languages like Amharic (Abate et al., 2018; Tsegay et al., 2020). Therefore, integrating robust morphological processing into Amharic NLP pipelines ensures that semantically related word forms are mapped to a shared underlying representation, improving both search relevance and document matching performance.

**Personal pronouns**

In Amharic, personal pronouns encode distinctions of person, number, and gender, which are crucial for semantic understanding and query matching. First-person singular "እኔ" (I) and plural "እኛ" (we) distinguish number, while second-person singular masculine "አንተ" (you) and feminine "አንቺ" (you) distinguish gender. Third-person pronouns further encode gender and number: "እሱ" (he), "እሷ" (she), and "እነሱ" (they). In a scanned document retrieval system, these pronouns help resolve references to subjects. For example, a query like *"Who wrote the book?"* should match both *"እሱ መጽሐፍ አነበበ"* (He read the book) and *"አልማዝ መጽሐፍ አነበበች"* (Almaz read the book). Proper identification of pronouns ensures high recall and semantic relevance by linking pronouns to their antecedents.

**Subject-Verb Agreement:**

Amharic verbs agree with their subjects in person, number, and gender, creating a variety of inflected forms. For instance, "አጻፍሁ" (I wrote) corresponds to first-person singular, "ተጻፈ" (It was written) marks passive voice, and "አጻፍን" (We wrote) marks first-person plural. Prefixes and suffixes indicate subject agreement: "አ-" for first-person singular, "-ን" for first-person plural, and "ተ-" for passive or reflexive forms. In scanned document retrieval, recognizing these agreements is essential. A query containing "አጻፍሁ" must also retrieve documents with "ተጻፈ" or "አጻፍን" to ensure completeness. Ignoring these agreements can result in false negatives due to verb inflection mismatches.

**Object Pronoun Suffixes:**

Amharic verbs often include suffixes indicating objects. For example, "አየኝ" (He saw me) combines the root "የ" (see) with the object pronoun suffix "-ኝ" (me). Rule-based retrieval systems must parse these object markers to normalize words and retrieve relevant documents accurately. For instance, a query like *"Who saw Almaz?"* should retrieve sentences like *"አየች*

*ኦልማዝ"*, even when the object suffixes vary. Morphological analyzers play a key role in reducing surface forms to canonical roots while preserving grammatical information.

**Possessive Suffixes:**

Amharic nouns attach possessive suffixes to indicate ownership. Examples include "ቤቴ" (my house), "ቤቷ" (her house), and "ቤታችን" (our house). This proposed retrieval systems must recognize these forms to correctly match queries like *"my book"* ("መጽሐፌ") to all relevant documents. Without handling possessive suffixes, a system may miss instances where different possessors are indicated, reducing the retrieval system's effectiveness.

**Reflexive Pronouns:**

Reflexive pronouns are formed by combining possessive suffixes with "ራስ" (self), resulting in forms such as "ራሴ" (myself), "ራሷ" (herself), and "ራሱ" (himself). These pronouns are common in narratives and administrative texts. NLP systems must resolve reflexive pronouns to their antecedents to ensure that queries about specific entities correctly retrieve all mentions, preserving context across sentences.

**Demonstrative Pronouns:**

Amharic distinguishes near and far demonstratives, as well as gender and number. Examples include "ይህ" (this, masculine), "ይህች" (this, feminine), "ያ" (that, masculine), and "ያች" (that, feminine). Correct handling of demonstratives allows retrieval systems to interpret queries like *"this book"* accurately, regardless of the form appearing in scanned documents.

**Nouns:**

Amharic nouns can be simple, e.g., "እግር" (foot), or derived, e.g., "እግር ጓደኛ" (pedestrian). Derived nouns often arise through vowel intercalation, affixation, or consonant modification. Effective scanned document retrieval requires recognition of both simple and derived nouns to avoid missing relevant content. For instance, a query for *"teacher"* should match both "መምህር" and its suffixed forms like "መምህሩ".

In general, Amharic grammar is a highly structured and morphologically rich system, where features such as pronouns, agreement markers, and affixation play a key role in meaning formation. In Amharic scanned document retrieval systems, understanding these grammatical structures is essential for normalizing different word forms and improving matching accuracy between queries and documents. Because a single Amharic root can generate many surface variations through

inflection and derivation, grammatical awareness helps reduce variation and improves retrieval performance in NLP applications.

This idea is strongly supported by prior work showing that morphological processing significantly improves NLP performance for Ethiopian languages by reducing word form variation and enhancing text matching (Abate et al., 2018).

### 2.2.5 Amharic Morphology

Amharic morphology is typified by non-concatenative, Semitic root and pattern processes, especially visible in verbs. Verbs generally derive from tri consonantal roots, into which vowels are interdigitated and affixes attached,(Amsalu & Demeke, 2006). For example, the root ገ ደ ለ (g d l) meaning "kill": its stem is "ገደል" (gə¬dəl), from which surface forms include "ገደልኩ" (gədəlku "I killed"), "ተገደለ" (təgədəl "he was killed") etc. SciTePress+1 The existence of multiple prefixes and suffixes (up to 4 prefixes and 5 suffixes per word) has been documented: "Learning Morphological Rules for Amharic Verbs" explains how authors trained an ILP system to extract rules of prefixes and suffixes. legacy.cs.indiana.edu In the context of scanned document retrieval for Amharic. NLP component must therefore perform morphological segmentation (prefix stem suffix), extract stems or roots, and enable query-matching across derived forms.

### 2.2.6 Amharic Attachments

Affixation in Amharic includes prefixes (e.g., causative "አ", reciprocal "ተ"), suffixes (plural "wot", object markers "ን", "ህ"), circumfixes and even infixes. For instance, the causative marker "a" and "as" may form causative derivations of passive stems, as in "Causative of the 'Passive' in Amharic". Lingua Franca For a document retrieval system, morphological analyzer must handle these affixes: strip or normalise them, map to base forms, and align variant derivations, so that retrieval queries can fetch the correct document segments even if affixed derivations appear in scanned text.

### 2.2.7 Compound Words in Amharic

Compound words in Amharic are formed by combining two or more free morphemes rather than using affixation. Examples include ቤተ ክርስቲያን (bete kristiyan, "church"), ትምህርት ቤት (timhirt bet, "school"), and ሥራ ቤት (sira bet, "office"). Amharic orthography is not fully standardized, and

such compounds may appear as separate words or inconsistently written forms in real texts, which creates challenges for NLP-based retrieval systems. Recent corpus-based studies of Amharic confirm that preprocessing and normalization including tokenization and handling orthographic variation are essential steps for improving consistency and retrieval performance in Amharic NLP applications (Yeshambel et al., 2022)

### 2.2.8 Punctuation in Amharic

Amharic uses punctuation symbols such as "።" (full stop), "፣" (comma), "፤" (semicolon), "፥" (colon), "፦" (preface colon) in addition to borrowed symbols "?" and "!" (Huet et al., 2007). In scanned documents especially, punctuation may be mis-recognized or omitted. In the retrieval pipeline, lexical analysis must account for these symbols in sentence segmentation and tokenization. The study "Amharic Ad hoc Information Retrieval …" mentions these punctuation marks and emphasizes their role in morphological and lexical processing.

### 2.2.9 Numerical digits in Amharic

Amharic uses Ethiopic numerals derived from the Geʿez script, representing numbers through additive symbols such as ፩ (1), ፲ (10), and ፻ (100), without a zero or place-value system. In modern texts, Arabic numerals are often used, but scanned historical documents still contain Ethiopic forms. So. document retrieval systems may normalize both Ethiopic and Arabic numerals to ensure consistent indexing and accurate query matching (Unicode Consortium, 2024).

## 2.3 Natural Language Processing

Natural Language Processing (NLP) is a branch of artificial intelligence and linguistics that enables computers to process, analyze, and understand human language (Jurafsky & Martin, 2023). In the context of Amharic scanned document retrieval, NLP provides the essential techniques for preprocessing, feature extraction, and semantic matching of text extracted from scanned documents.

### 2.3.1 Morphological Synthesis

Morphological synthesis is an essential component of natural language processing (NLP) systems, especially in languages with complex morphology like Amharic (Addis Ababa university 2010). A morphological synthesizer systematically generates all possible surface forms of a word from its lexical components, which are stored in a lexicon or database. This process is critical because

Amharic words undergo inflection and derivation based on tense, aspect, person, number, and gender. For instance, the root "ጻፍ" (säf, "write") can generate numerous forms such as "አጻፍሁ" (I wrote), "ተጻፈ" (it was written), and "አጻፍ" (imperative "write!"). In Amharic scanned document retrieval system, morphological synthesis ensures that all variations of a word can be matched to a single concept, allowing the system to retrieve documents accurately even when the surface forms differ. This approach significantly enhances recall by covering all word variants while maintaining precision in the retrieval results. Morphological synthesis also supports compound words and affixation, which are common in Amharic, thereby enabling the system to handle diverse linguistic structures (Perkiö et al., n.d.).

### 2.3.2 Morphological Analysis

Morphological synthesis is an essential component of natural language processing (NLP) systems, especially in languages with complex morphology like Amharic (Addis Ababa University, 2010). A morphological synthesizer systematically generates all possible surface forms of a word from its lexical components, which are stored in a lexicon or database. This process is critical because Amharic words undergo inflection and derivation based on tense, aspect, person, number, and gender. For instance, the root "ጻፍ" (säf, "write") can generate numerous forms such as "አጻፍሁ" (I wrote), "ተጻፈ" (it was written), and "አጻፍ" (imperative "write!"). In Amharic scanned document retrieval system, morphological synthesis ensures that all variations of a word can be matched to a single concept, allowing the system to retrieve documents accurately even when the surface forms differ. This approach significantly enhances recall by covering all word variants while maintaining precision in the retrieval results. Morphological synthesis also supports compound words and affixation, which are common in Amharic, thereby enabling the system to handle diverse linguistic structures (Jurafsky & Martin, 2023).

#### 2.3.3 Syntactic Analysis

Syntactic analysis examines the structural arrangement of words in a sentence, revealing grammatical relationships among sentence elements (Jurafsky & Martin, 2023). In Amharic, which generally follows a Subject–Object–Verb (SOV) order, parsing sentences accurately is essential for correct meaning interpretation. For example, in "አልማዝ መጽሐፍ አነበበች" ("Almaz read the book"), syntactic analysis identifies "አልማዝ" as the subject, "መጽሐፍ" as the object, and "አነበበች" as the verb. This process ensures that even with variations in sentence structure, a rule-based

scanned document retrieval system can correctly interpret relationships among words, improving the system's ability to match queries to relevant documents. syntactic analysis supports the handling of complex sentence structures, subordinate clauses, and compound sentences, which are common in natural language processing tasks and low-resource language processing systems

### 2.3.4 Semantic Analysis

Semantic analysis focuses on understanding the meaning of words, phrases, and sentences by considering their contextual usage and relationships within text. In Amharic NLP, semantic interpretation is particularly important because many words are context-dependent and can carry multiple meanings. For example, "ቤት" may refer to "house" in a general context or "school" depending on surrounding words. Semantic analysis therefore involves tasks such as word sense disambiguation, synonym recognition, and identifying semantic relationships between terms. In Amharic scanned document retrieval systems, semantic processing helps the system move beyond exact keyword matching and instead retrieve documents based on meaning equivalence. This improves retrieval accuracy and supports query expansion by identifying related terms that may not appear explicitly in the user's query (Jurafsky & Martin, 2023; Yimam et al., 2017).

### 2.3.5 Discourse Integration

Discourse integration analyzes meaning across multiple sentences by resolving references such as pronouns, anaphora, and context-dependent expressions. In Amharic, many expressions rely heavily on context, where pronouns refer back to previously mentioned entities. For example, in "አልማዝ መጽሐፍ አነበበች. እሷ በኋላ መጽሐፉን ተመልከተች" ("Almaz read the book. She later reviewed it"), the pronoun "እሷ" refers to "አልማዝ," and "መጽሐፉን" refers to the previously mentioned book. Incorporating discourse-level analysis in Amharic NLP systems improves document retrieval by correctly linking entities across sentences, reducing errors caused by surface-level mismatches and improving semantic continuity in scanned document retrieval systems (Jurafsky & Martin, 2023; Mitkov, 2002).

### 2.3.6 Pragmatic Analysis

Pragmatic analysis studies how language is used in context to interpret speaker intention, implied meanings, and contextual nuances. In Amharic NLP, pragmatics is important because the meaning of expressions can change depending on cultural and situational context. For example, the phrase "ትልቅ ግብር አለኝ" literally means "I have a big tax," but in context it may imply financial burden or

stress. Incorporating pragmatic analysis into Amharic scanned document retrieval systems allows the system to interpret idiomatic expressions and context-dependent meanings rather than relying only on literal word matching. This improves retrieval accuracy by aligning system output with intended meaning, especially in ambiguous or culturally grounded expressions (Jurafsky & Martin, 2023).

### 2.3.7 Lexical Analysis

Lexical analysis is the initial step in NLP, involving tokenization and text cleaning to prepare data for further processing (Jurafsky & Martin, 2023). Tokenization divides text into words, punctuation, and symbols, while text cleaning removes irrelevant characters such as special symbols and noise from scanned documents. In Amharic, this stage also includes handling orthographic variations, expanding abbreviations, and normalizing word forms that appear differently in scanned texts. For example, hyphenated or inconsistently written forms such as "መምህር-ቤት" may appear in documents and must be standardized during preprocessing. Effective lexical analysis ensures consistent indexing, improves query matching, and supports accurate retrieval in Amharic scanned document retrieval systems by providing clean and structured input for higher-level NLP tasks.

## 2.4 Empirical Literature Review

This section reviews existing research and practical applications related to document retrieval, Amharic NLP, and systems that address challenges with scanned documents. Under The following are essential point.

### 2.4.1 Information Retrieval from Scanned Documents

Information retrieval from scanned documents involves extracting, indexing, and searching textual content that is often produced through Optical Character Recognition (OCR). Traditional systems focus on improving indexing quality, layout analysis, and reducing noise introduced by OCR errors. Early approaches relied on simple keyword matching over OCR outputs, while more advanced methods incorporate probabilistic and language-model-based techniques to handle recognition uncertainty and noisy text. In particular, language modeling approaches in information retrieval improve robustness by estimating term probabilities and supporting more flexible document ranking under imperfect text conditions (Croft et al., 2009).

Additionally, OCR errors significantly affect retrieval performance, making post-OCR correction an important step. Techniques such as character-level language models, dictionary-based correction, and n-gram-based probabilistic models are commonly used to improve OCR output quality before indexing.

### 2.4.2 Applications of Amharic NLP

Despite being a low-resource language, some research efforts have been made in Amharic NLP:

- One important area is morphological analysis, where rule-based and finite-state approaches are used to handle the complex structure of Amharic words. Finite-state morphological techniques have been successfully applied to model Amharic word formation and improve linguistic processing (Gasser, 2010).
- Another important application area is named entity recognition (NER), where systems identify entities such as persons, locations, and organizations in Amharic text. Recent studies show that transformer-based multilingual models can be adapted effectively for Amharic NER, although performance is still limited by the lack of large annotated datasets (Devlin et al., 2019; Alemu et al., 2022).
- In addition, Optical Character Recognition (OCR) for Ethiopic script has been explored using machine learning and deep learning approaches. However, challenges such as font variation, degraded scans, and character complexity still affect recognition accuracy, making post-processing essential for downstream NLP tasks such as information retrieval (Abate et al., 2018).

### 2.4.3 NLP Systems in Practical

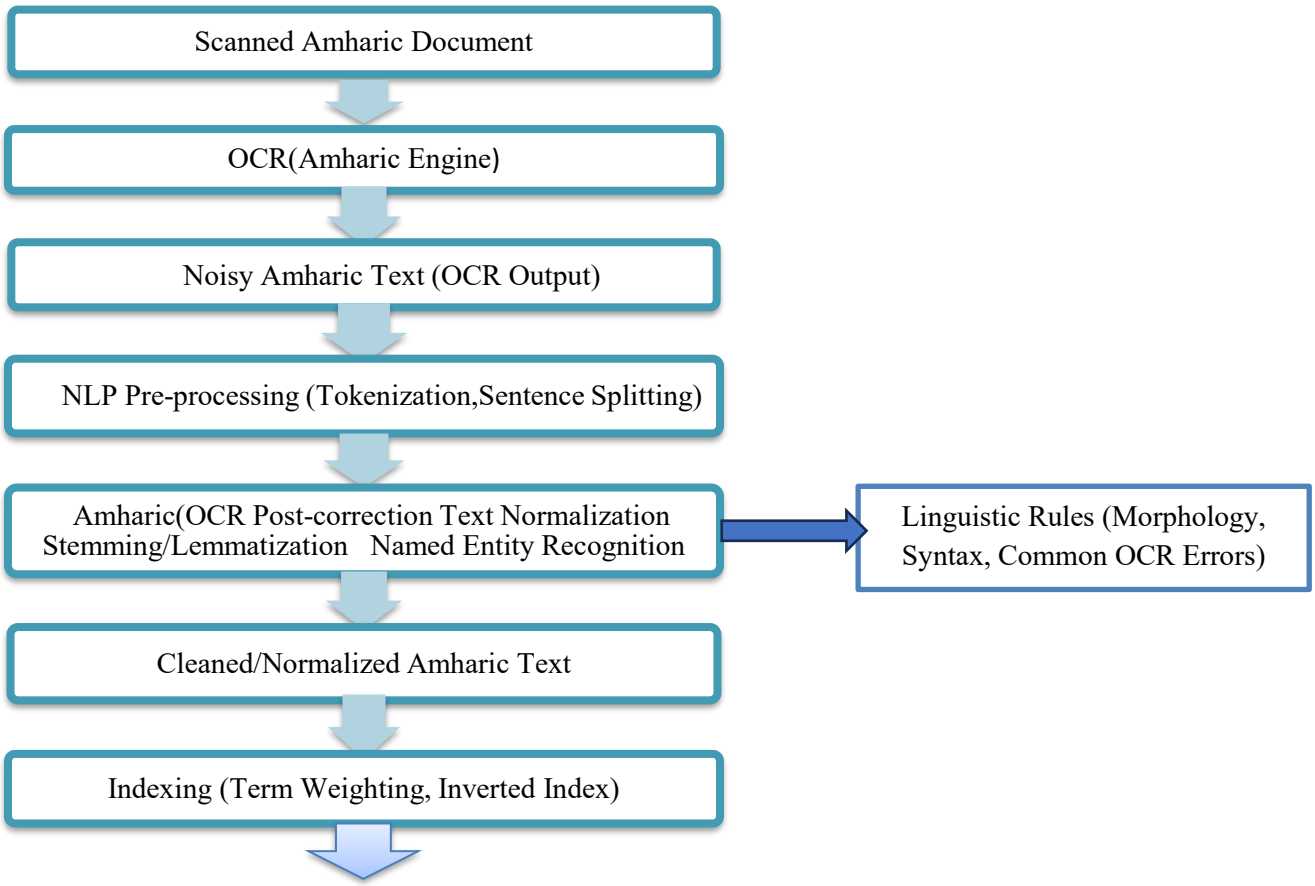
Early natural language processing (NLP) systems were mainly rule-based, relying on manually designed linguistic rules for tasks such as machine translation and language understanding. Systems like SYSTRAN used explicit grammatical and lexical rules to translate text, showing strong performance in limited and well-defined domains but struggling with ambiguity and scalability (Hutchins, 1986).

Modern NLP systems increasingly use hybrid approaches, combining rule-based linguistic knowledge with statistical and neural methods. Rule-based components help enforce grammatical structure, while data-driven models handle ambiguity and improve generalization across large datasets. This hybrid strategy is widely used in modern NLP systems to improve accuracy and robustness across different language processing tasks (Jurafsky & Martin, 2023).

## 2.5 Conceptual Framework

The conceptual framework illustrates the key components and their interactions within the proposed rule-based system for Amharic scanned document retrieval. It visualizes the flow of correction.

The framework proposes a pipeline approach, where each stage processes the input and prepares it for the next, with the component acting as the central intelligence for linguistic processing and error correction.



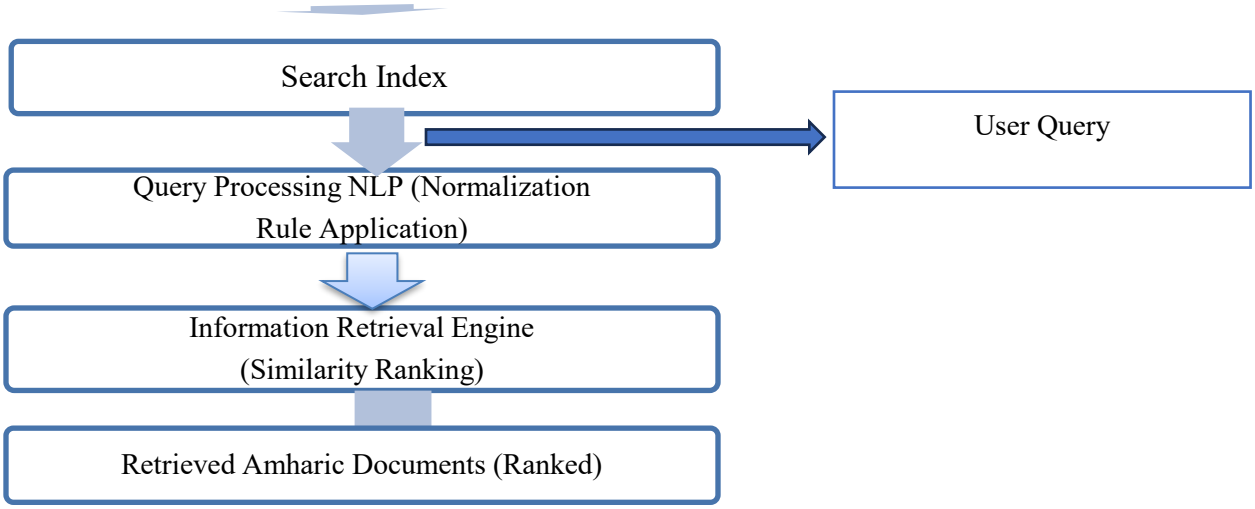



Figure 2.1 Conceptual Framework

**An explanation of a few of the components mentioned above**

- Scanned Amharic Document: The initial input, an image file (e.g., PNG, TIFF, JPEG) containing Amharic text.
- OCR (Amharic Engine): A pre-existing Amharic OCR software that converts the image into raw, potentially noisy, digital text.
- Noisy Amharic Text (OCR Output): The primary data that the proposed system will work with.
- NLP Pre-processing: Basic text cleaning operations such as tokenization (splitting text into words/units) and sentence splitting.
- Rule-Based Amharic Text Normalization: This is the core component of the proposed system. It applies a set of predefined rules to:

This framework highlights the central role of NLP in transforming noisy OCR output into a format suitable for accurate and efficient information retrieval for Amharic documents.

# CHAPTER THREE

# THEORETICAL BACKGROUND OF DOCUMENT RETRIEVAL

This chapter introduces the key concepts and theoretical background of document retrieval that support the proposed system. It explains the main components of information retrieval, including retrieval models, indexing techniques, and natural language processing methods. The chapter also highlights important linguistic challenges in Amharic, such as morphological variation and orthographic inconsistency, which can affect retrieval performance.

## 3.1 Information Retrieval

Information Retrieval (IR) is a well-established field in computer science and information science that studies how users' express information needs and how systems retrieve relevant documents from large collections of unstructured or semi-structured data. These collections may include digital libraries, web pages, institutional repositories, and scanned document archives. The primary goal of an IR system is to return documents that are relevant to a user's query and to rank them according to estimated usefulness, so that the most important results are presented first (Manning et al., 2008).

A typical IR system consists of several core components: document preprocessing, indexing, query processing, and ranking. In preprocessing, documents are cleaned and transformed into a structured representation. Indexing then builds efficient data structures (such as inverted indexes) that allow fast searching. Query processing interprets the user's request and converts it into a comparable representation, while ranking algorithms compute relevance scores between the query and documents using statistical or probabilistic models.

Early IR models, such as Boolean retrieval, treated documents as sets of keywords and relied on exact matching between query terms and document terms. Although simple, this approach lacks the ability to rank documents effectively or handle linguistic variation. To overcome these limitations, probabilistic and vector space models were introduced, allowing partial matching and relevance ranking based on term frequency and document similarity. These models significantly improved retrieval effectiveness and became the foundation of modern search engines (Croft et al., 2009).

A key challenge in IR is the ambiguity and variability of natural language. Words can have multiple meanings depending on context, and the same concept can be expressed in different ways. This problem becomes more severe in large-scale collections where users expect precise and fast results. As a result, modern IR systems incorporate Natural Language Processing (NLP) techniques such as tokenization, stemming, lemmatization, and semantic analysis to improve both query understanding and document representation.

To address these challenges, modern information retrieval (IR) systems incorporate Natural Language Processing (NLP) techniques to improve document representation, query interpretation, and relevance ranking. NLP enables systems to move beyond simple keyword matching by considering linguistic features such as morphology, syntax, and semantics, allowing for more accurate understanding of both queries and documents. These enhancements are particularly important for handling language variability and improving retrieval effectiveness in complex text environments (Manning et al., 2008; Croft et al., 2009)

In the context of my thesis Amharic scanned document retrieval using NLP techniques serve as the backbone for correcting spelling variations, reducing morphological complexity, normalizing word forms, and improving the reliability of OCR-derived text. Despite these improvements, IR systems still face challenges due to the complexity of natural language. These basic language problems must be addressed to achieve accurate retrieval, especially for morphologically rich languages like Amharic. The next section highlights the key language-related issues affecting IR systems.

### 3.1.1 Basic Language Problems in Information Retrieval

Natural languages present a wide range of complexities that directly affect the performance of information retrieval systems. These difficulties are especially prominent in Amharic because of its unique writing system, morphological richness, and non-standardized orthographic practices. When scanned Amharic documents are processed through OCR, inconsistencies become even more problematic, leading to mismatches between query terms and document terms. To build a rule-based Amharic retrieval system, it is essential to recognize and address the major linguistic issues that hinder IR performance. Among these, the main challenges include morphological variation, orthographic inconsistency, and structural complexity, which significantly affect accurate indexing and retrieval

**Morphology**

Morphological variation is one of the most challenging linguistic features affecting IR effectiveness, particularly in Semitic languages such as Amharic. Words may appear in a wide array of inflected or derived forms. For example, verbs encode subject markers, object markers, tense, aspect, and mood, all embedded within the word. Nouns may carry plural markers, definiteness markers, and possessive suffixes. Compounds and derivative patterns further increase the number of surface forms. As a result, a simple keyword match fails to identify semantically equivalent words.

In English text processing, stemming techniques such as Porter's algorithm help reduce different word forms to a common root, which improves information retrieval by minimizing variation in word structure (Porter, 1980). In modern NLP, this idea has evolved further. Instead of relying only on rule-based stemming, many systems now use lemmatization and transformer-based models, which learn word meanings from context and often handle morphological variation more effectively (Jurafsky & Martin, 2023).

Amharic presents a greater challenge because its word formation is based on a root-and-pattern system rather than simple prefix or suffix changes. This means that a single root can generate many different word forms depending on grammatical structure. As a result, Amharic information retrieval systems need explicit morphological processing to identify roots, normalize word forms, and connect different variations of the same meaning.

### 3.1.2 Query

Queries represent the user's expression of their information need, and in an information retrieval (IR) system, they undergo the same type of linguistic processing as the document collection. This is particularly important in Amharic scanned document retrieval, where OCR-generated text may contain noise, spelling variations, and inconsistencies that must be aligned with document preprocessing standards.

In vector space models, the terms extracted from a user query are transformed into a mathematical representation, typically a vector in a high-dimensional term space, allowing comparison with document vectors (Manning et al., 2008). Each term is assigned a weight based on its importance in the query and its statistical distribution in the corpus, such as term frequency and inverse document frequency (TF-IDF). The system then computes similarity between the query vector and

document vectors, commonly using cosine similarity, to measure how closely a document matches the user's intent. Documents are then ranked in descending order of similarity and presented as the final retrieval output.

The effectiveness of query processing is also influenced by query length. Longer queries provide richer contextual information, which improves the performance of NLP techniques such as morphological analysis and disambiguation. In contrast, short queries often lack sufficient context, making it more difficult for the system to accurately interpret user intent. Therefore, in Amharic IR systems, more descriptive and structured queries generally lead to better retrieval performance (Croft et al., 2009).

In the context of my thesis, query processing involves applying NLP procedures such as:

- normalization of spelling variants,
- decomposition of attached pronoun markers,
- morphological reduction to base forms,
- filtering of Amharic stop-words,
- correction of OCR-induced errors.

By ensuring that queries undergo the same linguistic normalization as documents, the retrieval system achieves higher precision and recall.

### 3.1.3 Natural Language Processing in Information Retrieval

Natural Language Processing (NLP) is essential for building effective information retrieval (IR) systems, especially for morphologically rich languages such as Amharic, where linguistic complexity strongly affects indexing, searching, and ranking. NLP techniques enable systems to extract deeper linguistic features from text and convert unstructured documents into structured representations that are more suitable for retrieval and analysis (Jurafsky & Martin, 2023).

In Amharic scanned document retrieval systems, NLP is applied immediately after Optical Character Recognition (OCR) to clean, normalize, and linguistically process the extracted text before indexing. This step is important because OCR output often contains noise, spelling inconsistencies, and structural errors that reduce retrieval accuracy. By applying NLP preprocessing, the system improves text quality and ensures better alignment between queries and documents.

In this research, NLP is applied directly after OCR extraction to normalize and refine the text before indexing. The following steps describe the main NLP preprocessing activities that enhance the accuracy and efficiency of retrieving Amharic scanned documents.

**Stop Words Removal**

Stop words are extremely common terms that appear frequently across documents but contribute little to distinguishing one document from another. These include determiners, prepositions, conjunctions, and other function words. Removing them reduces computational load and improves retrieval efficiency. For Amharic, stop-word removal is complex because many function words appear as prefixes or suffixes attached to content words. This requires rules and NLP decomposition before stop-word filtering is applied. Using frequency statistics and linguistic rules, IR systems discard these terms during indexing to improve performance and reduce noise (Manning et al., 2009).

**Stemming**

Stemming reduces different forms of a word to a common base form. This process is crucial for IR systems because morphological variants often express the same underlying meaning. For example, families of related words, such as democrat, democracy, and democratic, can all map to a shared stem. In Amharic, stemming require rule because its morphology involves root-and-pattern structures rather than simple suffix addition. Stemming is applied during both indexing and query processing, ensuring that equivalent forms are matched accurately, reducing vocabulary size, and improving retrieval effectiveness (Porter, 1980).

**Part-of-Speech Tagging**

Part-of-speech (POS) tagging labels each word in a sentence with its grammatical role, such as noun, verb, or adjective. This helps NLP systems reduce ambiguity and supports tasks like lemmatization and text filtering (Jurafsky & Martin, 2023).

In Amharic, POS tagging is especially important because words often contain rich morphological information within a single form. It helps separate meaningful content from grammatical markers and improves text understanding, particularly in noisy OCR-generated documents where spacing and punctuation may be unreliable. POS tagging is especially beneficial when dealing with OCR-derived text, where the system must rely on linguistic rules rather than perfect spacing or punctuation.

### Chunking and Shallow Parsing

Chunking groups words into meaningful phrases such as noun phrases and verb phrases, allowing NLP systems to capture more meaningful units than individual words. It is a form of shallow parsing that extracts partial syntactic structure without performing full syntactic analysis, making it efficient for large-scale text processing (Jurafsky & Martin, 2023).

In Amharic scanned document retrieval, chunking is useful for identifying important multi-word expressions such as names of people, places, and compound terms. It also helps correct or reduce issues caused by OCR errors where words may be incorrectly split or merged, improving the quality of information extraction and retrieval.

### Additional NLP Processes

Other important NLP processes in information retrieval include compound splitting, identification of head modifier relationships, normalization of orthographic variations, and detection of multi-word expressions. These techniques help improve the structural and linguistic representation of text before indexing and retrieval (Manning et al., 2008; Croft et al., 2009).

In Amharic scanned document retrieval systems, these processes are particularly important due to OCR noise, rich morphological structure, and inconsistent spelling variations in real documents. Applying NLP-based normalization and linguistic rules helps ensure that different surface forms are mapped to consistent representations, improving retrieval accuracy and system reliability.

### 3.1.4 Document Indexing

Document indexing is a core process in information retrieval (IR) systems that enables efficient organization, storage, and retrieval of large document collections. It transforms raw text into structured representations that can be quickly searched and matched with user queries. In Amharic scanned document retrieval systems, indexing is particularly important because OCR output often contains noise, spelling variations, and morphological complexity that must be normalized before effective retrieval can take place.

Indexing typically works by extracting meaningful terms or features from documents and organizing them into efficient data structures. The most widely used structure is the inverted index, which maps each term to the list of documents in which it appears. This allows the system to quickly locate relevant documents without scanning the entire collection, making it highly efficient for large-scale search tasks (Manning et al., 2008; Croft et al., 2009).

In addition to traditional indexing, modern IR systems increasingly combine classical methods with neural approaches. For example, transformer-based models such as BERT can represent documents as dense vectors that capture semantic meaning beyond exact keyword matching. However, inverted indexes are still widely used because they provide fast and scalable retrieval, especially in large document collections. In practice, many modern systems use a hybrid approach that combines both sparse (inverted index) and dense (neural embedding) representations to improve accuracy and relevance (Devlin et al., 2019).

In large-scale information retrieval systems such as web search engines, indexing plays a crucial role in determining the speed, accuracy, and scalability of retrieval. A well-designed index ensures that relevant documents can be efficiently retrieved, while a poorly designed one can negatively affect system performance. For example, overly limited indices may fail to capture important terms or linguistic cues, while overly large indices may introduce noise and redundancy, reducing search efficiency and accuracy (Manning et al., 2008).

In traditional IR systems, documents are represented using index terms generated through preprocessing steps such as tokenization, normalization, and stemming. These index terms typically represent meaningful lexical units that capture the core content of a document. The indexing process can occur during document ingestion (index-time) or be influenced during query processing (search-time), depending on system design. The indexer component is responsible for analyzing document content, removing irrelevant tokens, extracting important features, and organizing them into efficient data structures such as inverted indexes for fast retrieval (Croft et al., 2009.

In this Amharic retrieval system, the indexer may also apply Amharic-specific linguistic rules such as root-pattern analysis, normalization of orthographic variants (e.g., ሀ/ሐ/ኀ equivalence), and segmentation of compound verbs, making the indexing pipeline particularly language-aware.

Indexing in information retrieval systems typically involves several conceptual phases. First, meaningful units or keywords are extracted from the raw text. Second, the linguistic structure surrounding these units is considered to understand how words relate to each other in context. Third, the functional role of words is analyzed to capture their meaning within sentences and phrases. These steps help ensure that documents are represented in a way that supports accurate and efficient retrieval.

Based on these processes, different types of indices can be constructed, including lexical indices (based on normalized word forms), morphological indices (based on stems or roots), and subword or token-based indices that capture finer linguistic structures. These multiple levels of representation improve retrieval performance, especially in languages with rich morphological variation. (Manning et al., 2008; Croft et al., 2009).

eature selection is a key step in the indexing process of information retrieval (IR) systems. It involves identifying which terms from a document should be included in the index so that the document is accurately and efficiently represented. In modern IR systems, term importance is often estimated using statistical signals such as term frequency and inverse document frequency (TF-IDF), which help distinguish meaningful words from less informative ones (Manning et al., 2008).

The feature selection process generally includes three main stages. First, preprocessing is applied to clean the text, including tokenization, stop-word removal, and normalization. Second, statistical measures such as term frequency are computed to evaluate how important each term is within a document and across the collection. Finally, the most informative terms are selected to form the document representation used in indexing. These steps reduce noise and improve retrieval efficiency by focusing on relevant linguistic features.

In recent IR and NLP systems, feature selection has been further enhanced using neural and subword-based representations. Transformer models such as BERT reduce reliance on manual feature selection by learning contextual word representations directly from data. However, for low-resource and morphologically rich languages such as Amharic, explicit preprocessing and normalization are still important to handle spelling variation, affixation, and root-based morphology effectively. As a result, many practical systems combine traditional statistical methods with modern neural approaches to improve retrieval quality (Devlin et al., 2019; Jurafsky & Martin, 2023).

There are various types of indexing techniques that can be applied, including inverted indexing, signature files, term-document matrices, and n-gram indexing, each designed to improve the efficiency and accuracy of document retrieval.

**Signature Files:**

This method creates a compact "signature" or fingerprint for each document, usually in the form of a binary string. These signatures allow the retrieval system to quickly eliminate irrelevant documents before performing more detailed comparisons, improving search efficiency in large collections.

**N-gram Indexing:**

N-gram indexing breaks text into sequences of *n* characters or words, called n-grams. This technique is particularly useful for handling word variations, misspellings, and morphological differences, which makes it highly suitable for languages like Amharic with rich morphological structures. N-gram indexing can improve both recall and precision in retrieval tasks.

**Inverted File**

The inverted file, commonly known as the inverted index, is the most widely used indexing structure in operational information retrieval (IR) systems, including text-based search engines and systems that process OCR-derived documents. It is especially effective in Amharic scanned document retrieval because it enables fast matching between processed linguistic terms and the documents in which they occur.

The main idea of the inverted index is simple. Each document in a collection is represented by a set of meaningful terms, such as keywords or normalized words, that describe its content. These terms are stored in an inverted structure where each term points to a list of documents containing it. This allows the system to quickly retrieve relevant documents without scanning the entire collection. In modern IR systems, this approach is extended to full-text indexing, where most meaningful terms in a document are included in the index to improve retrieval coverage and effectiveness (Manning et al., 2008; Croft et al., 2009).

Before indexing, preprocessing steps such as stop-word removal, token normalization, and stemming or lemmatization are applied to reduce noise and unify word forms. These steps are particularly important for Amharic due to its rich morphology and orthographic variation, including multiple representations of the same phoneme (e.g., ሀ/ሃ/ሐ/ኀ). Without normalization, these variations can reduce matching accuracy and increase index redundancy.

Stemming is an important text normalization technique used in information retrieval (IR) systems to reduce different forms of a word into a common base form. The main objective is to improve retrieval performance by grouping related word variations under a single representation, allowing the system to match user queries and documents more effectively. In addition, stemming helps reduce vocabulary size and improves indexing efficiency in large document collections (Jurafsky & Martin, 2023).

**Suffix Tree**

A suffix tree is another powerful indexing structure used primarily for string processing tasks. It is particularly useful for applications that require efficient substring matching, approximate matching, or structural decomposition of text. While less commonly used for large-scale IR than inverted files, suffix trees are valuable in special contexts, such as pattern-based retrieval or linguistic segmentation both of which are relevant to rule-based Amharic NLP.

A suffix tree is an efficient tree-based data structure used for fast pattern matching and substring searching within large text collections. It stores all suffixes of a text in a compact trie-like structure, allowing rapid search operations and efficient identification of repeated or related character patterns. Because of these capabilities, suffix trees are widely applied in modern information retrieval, text mining, and computational linguistics (Jurafsky & Martin, 2023).

Recent studies on compact data structures and string indexing also highlight the importance of suffix-tree-based methods for handling large-scale and noisy textual data efficiently (Navarro, 2022). In information retrieval systems, suffix trees support approximate matching, partial-word searching, and pattern discovery, which are especially useful in OCR-based retrieval environments where text may contain recognition errors or inconsistent spellings.

For Amharic scanned document retrieval, suffix trees can help identify morphological patterns, repeated character sequences, and OCR-generated variations. This is particularly important because Amharic uses a syllabic writing system with many visually similar fidel characters that may be confused during OCR processing. Instead of relying only on exact term matching, suffix-tree-based matching can improve retrieval robustness by detecting structurally related word forms and reducing the impact of spelling inconsistencies and OCR noise.

### 3.1.5 Term Weighting

Term weighting is a fundamental component of information retrieval (IR) systems because it directly influences how documents are ranked and retrieved in response to user queries. In the context of Amharic scanned document image retrieval, term weighting becomes even more essential, as Amharic documents present unique linguistic complexities, including rich morphology, large inflectional variation, and frequent orthographic inconsistencies. After a scanned Amharic document is processed either through OCR or other character recognition its textual features must be represented in a form that allows the retrieval engine to judge how relevant the document is to a user's query. Term weighting provides this mechanism by assigning a numerical importance score to each term within each document, thereby enabling the retrieval model to distinguish between highly informative terms and those that occur frequently but carry minimal semantic significance.

Term weighting is an important component of information retrieval (IR) systems because it helps determine how important a term is within a document and across the entire document collection. Different IR models use different weighting approaches to improve document ranking and retrieval effectiveness.

In probabilistic IR models, term weighting methods estimate the likelihood that a term appears in relevant documents compared to non-relevant ones. These probability-based approaches help the system rank documents according to their estimated relevance to a user query. In contrast, vector space models rely more on statistical and empirical evidence obtained through system evaluation and retrieval experiments to determine term importance (Manning et al., 2008).

Most modern term weighting techniques are based on three main factors: term frequency (TF), document frequency (DF), and document length. Term frequency measures how often a term appears in a document, while document frequency indicates how common or rare the term is across the entire collection. Document length is also considered because longer documents naturally contain more terms than shorter ones. Together, these factors help identify informative terms and improve ranking accuracy in IR systems.

Term frequency (tf) refers to how often a term appears within a document. The intuition behind term frequency is that a term occurring multiple times is likely to be more representative of a document's topic. In Amharic, this factor must be handled carefully since different surface forms

of a root (e.g., ተመለሰ፣ መመለስ፣ ተመላለሰ) may inflate frequency counts unless normalized through rule-based morphological processing. Document frequency (df), on the other hand, refers to the number of documents in which a term appears. A term that occurs in many documents is less discriminative and therefore less helpful for identifying specific document content. To address this, IR systems commonly use the inverse document frequency (idf) measure, which gives higher weight to rare terms and lower weight to common terms. This concept is especially important in Amharic text retrieval because certain functional morphemes and auxiliary constructions are common across many documents and must be down-weighted to prevent them from overshadowing more content-rich words.

Another important idea in information retrieval is document length normalization. In real collections, documents are rarely the same size. Some may be short news articles, while others can be long Amharic law books, religious texts, or newspaper archives. If we do not adjust for this difference, longer documents will naturally have higher term frequencies simply because they contain more words. This can unintentionally push them higher in search results, even if they are not more relevant than shorter documents.

To avoid this problem, length normalization is used to balance the influence of document size when calculating relevance scores. It ensures that a document is not ranked highly just because it is long, but because it truly matches the user's query. In this way, retrieval systems produce fairer and more accurate results across documents of different lengths (Zhai & Massung, 2016; Büttcher et al., 2016).

Mathematically, term frequency is denoted by $tf_{t,d}$, representing the number of times term *t* appears in document *d*. Document frequency, denoted by $df_t$, corresponds to the number of documents containing term *t*. Inverse document frequency is then computed using the formula:

$$idf_t = \log\left(\frac{ND}{ND_i}\right)$$

Where:

ND is the total number of documents in the collection

ND i is the number of documents in which term *i* appears

A rare term receives a higher inverse document frequency (IDF) value, which reflects its stronger ability to distinguish relevant documents from non-relevant ones. A widely used approach that

combines both term frequency (TF) and inverse document frequency (IDF) is the TF-IDF weighting scheme. This method assigns each term a combined score that increases when a term is frequent in a specific document but rare across the entire collection. In this way, TF-IDF helps retrieval systems emphasize terms that are both important within a document and unique across the dataset, improving overall ranking effectiveness (Manning, Raghavan, & Schütze, 2008). The tf-idf formula is expressed as:

$$tf\text{-}idf = tf_{t,d} \times idf_t$$

In retrieval, scoring determines how strongly a document matches a user's query. If a document contains a query term multiple times, its score increases proportionally. The final similarity score between a query and a document is therefore directly influenced by the weighted representation of the query terms in that document. In proposed Amharic IR, this weighting process becomes stronger when paired with linguistic normalization strategies such as root extraction, homophone resolution (e.g., normalizing ሐ, ሀ, ኀ), and morphological decomposition, allowing tf-idf to operate on clean and semantically meaningful Amharic units.

In general, term weighting is not merely a mathematical computation but a critical stage that bridges the linguistic structure of Amharic text with the statistical foundations of modern IR. It ensures that the retrieval system can effectively differentiate between meaningful content and noise an especially important requirement in scanned Amharic document retrieval where OCR imperfections and spelling inconsistencies often challenge the system's ability to rank documents accurately.

### 3.1.6 Ranking

Once the system identifies all documents containing terms similar to the user's query, the next critical step is to determine which documents are most relevant. This is accomplished through a ranking mechanism. In the proposed Amharic Document Image Retrieval System (ADIRS), ranking is based on term frequency (TF), one of the simplest yet highly effective measures in classical information retrieval. The underlying principle of TF is that the more often a query term appears in a document, the more representative that document is of the user's information need. Due to its interpretability and suitability for word-level image retrieval, TF is adopted as the primary ranking algorithm in this study (Baeza-Yates & Ribeiro-Neto, 2011).

The normalized term frequency for a term t in a document d is computed as:

$$\text{Tf(t,d)}=\frac{n(t,d)}{\sum_k n(k,d)}$$

Where n(t,d)n(t,d)n(t,d) is the number of occurrences of term ttt in document ddd, and ∑kn(k,d)\sum_{k} n(k,d)∑kn(k,d) is the total number of words in the document. Normalization ensures that the relevance score is independent of document length, allowing fair comparison across documents of different sizes.

During the ranking stage, the system first processes the user's query by breaking it down into its root forms using the NLP and stemming components of ADIRS. Each query term is then compared with the entries stored in the inverted index. If the query contains multiple words, the system retrieves the postings list for each term and combines them into a single set of candidate documents.

For each candidate document, the system checks how often each query term (or its related form) appears. These values are then used to compute a relevance score based on term frequency (TF), which gives an idea of how closely a document matches the user's query.

Lastly, the system sorts all candidate documents in descending order of their TF scores. Documents that contain more occurrences of the query terms are placed higher in the result list, since they are more likely to match the user's information need. this approach works well as a baseline ranking method for Amharic document image retrieval systems and also forms the foundation for more advanced models like BM25 (Manning, Raghavan, & Schütze, 2016; Robertson & Zaragoza, 2009).

### 3.1.7 Documents Database

The documents database represents the central storage component of the information retrieval system. It contains all items whether text files, scanned document images, metadata, or extracted linguistic features that have been received, processed, indexed, and preserved for retrieval. In Amharic scanned document retrieval environment, this database serves as the core knowledge source from which the system draws information whenever a user submits a query. Every query is evaluated against the full collection of documents or their indexed representations, ensuring that the retrieval process is exhaustive and consistent.

Traditional database management systems (DBMS) excel at handling structured data are capable of returning highly precise results when queries are expressed using formal, well-defined syntax. However, IR systems differ significantly in those users often express queries using vague, incomplete, or natural language phrasing. For Amharic users in particular, queries may vary widely due to dialectal differences, spelling variations, or differences in morphological form. Since DBMS frameworks typically require strictly defined queries, they cannot directly support flexible, linguistically complex query interpretation. Therefore, the IR system must be designed to process and interpret natural-language-based queries effectively and to translate them into meaningful search operations.

The documents database in IR systems is, therefore, not merely a storage space it is a semantically rich representation of textual material prepared for intelligent retrieval. It enables the system to compare the linguistic patterns extracted from a user query with the patterns embedded in stored documents. In systems using Amharic NLP approaches, this database often stores normalized root forms, token sequences, and morphological structures to ensure consistency. In this way, the document database functions as both the backbone of the retrieval process and the repository from which linguistic evidence is drawn to identify and rank relevant documents.

### 3.1.8 Information Retrieval Models

Information retrieval systems rely on conceptual models to formally represent documents, queries, and the process of matching them. These models define how documents are described, how relevance is estimated, and how ranking is performed. In Amharic scanned document retrieval, IR models must be adapted to handle the rich morphology and orthographic complexity of the Amharic language, making the choice of retrieval model a critical design decision. Three major retrieval models are commonly used: the Boolean Model, the Vector Space Model, and the Probability Model. An additional model, the Inference Network Model, provides a more flexible framework for representing relationships between terms and documents.

#### Boolean Model

The Boolean retrieval model is one of the earliest approaches in information retrieval. It uses logical operators such as AND, OR, and NOT to combine query terms into structured expressions that define exact matching conditions (Hirpassa, 2024). In this model, documents are treated as sets of words, and retrieval is based on whether a document satisfies the query's logical rules.

For example, a query like “education AND Amharic NOT history” returns documents containing both “education” and “Amharic” while excluding those with “history.” These operations correspond to basic set theory concepts such as intersection, union, and difference. Although simple and easy to implement, the Boolean model does not support partial matching or result ranking, which limits its use in large-scale retrieval systems (Baeza-Yates & Ribeiro-Neto, 2011).

While Boolean retrieval allows users to formulate highly structured and specific queries, it has several limitations. There is no inherent mechanism for ranking results; documents either match the query or do not. As a result, the model cannot distinguish between documents that barely satisfy the query and those that are highly relevant. For Amharic document retrieval, this limitation is even more pronounced because linguistic variability affects how terms appear in documents. Users may struggle to construct Boolean queries that account for Amharic morphological variations, homophones, or orthographic inconsistencies. Nonetheless, Boolean retrieval can still be useful in certain rule-based environments where precise filtering is needed, such as selecting documents based on metadata or exact keyword matches.

**Vector Space Model (VSM)**

The Vector Space Model is one of the most widely used frameworks in modern IR systems because it enables graded relevance ranking rather than binary decision-making (Argaw & Asker, n.d.). In this model, both documents and queries are represented as vectors in an N-dimensional space, where N corresponds to the total number of indexing terms in the system. Each dimension represents a specific term, and the value in each dimension corresponds to the weight of that term typically computed using TF–IDF.

The Vector Space Model (VSM) supports free-text queries, allowing users to enter one or more terms without requiring strict Boolean syntax (Manning, Raghavan, & Schütze, 2008). This flexibility is particularly useful for Amharic document retrieval, where users often input natural language queries and where NLP techniques can normalize morphological variations before vector representation. In this model, relevance is determined by measuring the similarity between the query vector and each document vector. This is commonly done using cosine similarity, which calculates the angle between two vectors and produces a score between 0 (no similarity) and 1

(identical vectors). Based on these similarity scores, documents are ranked from most to least relevant to the user's query

If D represents a document vector and Q represents a query vector, the similarity is computed as:

$$\text{Sim}(\vec{D}, \vec{Q}) = \sum_{ti \in D,Q} WtiQ.WtiD$$

Where $W_{ti}^{Q}$ and $W_{ti}^{D}$ represent the weights of term $t_i$ in the query and document vectors, respectively. In Amharic scanned image retrieval systems, linguistically normalized terms such as root forms, stemmed variants, and orthographically harmonized words help ensure that VSM accurately captures semantic similarity even when surface forms differ.

**Probability Model**

The probabilistic model in information retrieval ranks documents based on their estimated probability of relevance to a user's query, rather than relying on exact matching or vector representation. It assumes that each document has a certain likelihood of satisfying the user's information need, which is estimated using term statistics from the document collection (Baeza-Yates & Ribeiro-Neto, 2011). Since true probabilities cannot be directly measured, the model uses approximations such as term distributions in relevant and non-relevant documents. This idea is formalized in the Binary Independence Model and forms the theoretical basis for modern ranking approaches like BM25, which improves ranking by incorporating term frequency and document length normalization (Robertson & Zaragoza, 2009).

A document's probability of being relevant is expressed as $P(R \mid D)$, and non-relevance as $P(\bar{R} \mid D)$. Documents are ranked by decreasing order of:

$$\log\left(\frac{P(R|D)}{P(\bar{R}|D)}\right)$$

This ratio expresses how much more likely the document is relevant compared to irrelevant. In Amharic IR, probabilistic models can be powerful, but they rely heavily on accurate linguistic preprocessing. If a term appears in many documents due to morphological variants or orthographic variation, the probability model must rely on normalization to avoid inaccurate probability estimates.

**Inference Network Model**

The Inference Network Model treats information retrieval as a probabilistic inference process, where documents, terms, and queries are represented as interconnected nodes in a network structure. In this framework, a document provides evidence for query terms, and this evidence is interpreted as a weighted contribution indicating how strongly the term represents the document's content. These individual pieces of evidence are then combined across the network to compute an overall relevance score for ranking documents. This model is part of a broader class of probabilistic retrieval approaches that aim to capture uncertainty and dependencies between query terms and documents. systems (Croft, Metzler, & Strohman, 2015; Manning, Raghavan, & Schütze, 2016).

One advantage of this model is its flexibility; it does not prescribe a fixed method for determining the strength of term instantiation. This makes it suitable for this Amharic retrieval systems, were linguistic rules such as root extraction, pattern matching, or morphological normalization can define term weights according to language-specific criteria. In practice, the ranking produced by an inference network often resembles that of the vector space or probabilistic models, but it provides a more unified and logically principled framework for combining linguistic evidence.

### 3.1.9 Evaluation Techniques for IR Systems

In any Information Retrieval (IR) system, including those designed for Amharic scanned document image retrieval using NLP techniques, evaluating system performance is a crucial step. Although different retrieval models and indexing strategies may be used, the ultimate question remains the same: How effectively does the system locate the information that users actually need? To answer this, IR research has established standard evaluation procedures based on the concepts of relevance, retrieval quality, and system effectiveness.

Evaluation becomes even more important in the context of Amharic document image retrieval, because Amharic is morphologically rich, uses the Ge'ez script, and scanned images often suffer from variations in font style, noise, degradation, and handwritten or printed inconsistencies. The proposed NLP approach relies heavily on accurate or linguistic cues so the system's effectiveness must be measured rigorously to ensure that the handcrafted rules perform correctly under different document conditions.

The core idea in information retrieval evaluation is to determine whether retrieved documents are relevant or non-relevant to a user's query. Relevance is represented by a score that estimates how

well a document matches the user's information need. Higher scores indicate stronger relevance, while lower scores indicate weak or no relevance. In general, relevant documents provide useful information that satisfies the query, whereas non-relevant documents do not contribute meaningfully to the search result (Büttcher, Clarke, & Cormack, 2016).

When a user issues a query such as searching for an Amharic word, phrase, or morphological pattern the entire document database can be logically divided into four categories, often illustrated in IR literature. These categories reflect the interaction between relevance and retrieval:

**Effects of Search on Document Space**

- Relevant and Retrieved
- Relevant but Not Retrieved
- Not Relevant but Retrieved
- Not Relevant and Not Retrieved

This partitioning is typically represented as the "search outcome space," which visualizes how a system's retrieval operations affect document classification.

To measure how well the system performs within this space, several widely accepted evaluation metrics are used. These metrics help determine how effectively the system retrieves Amharic document images that truly match the linguistic or structural features specified in a query.

**Recall**

Recall measures the ability of the IR system to retrieve all relevant documents. In the context of Amharic NLP-based image retrieval, this might include detecting all scanned pages containing a particular word shape, stem pattern, suffix, or linguistic feature. Recall specifically evaluates the proportion of relevant documents that the system successfully retrieves.

Formally, recall is defined as:

$$\text{Recall (R)} = \frac{\text{Number of Relevant Items Retrieved}}{\text{Total Number of Relevant Items}}$$

A high recall indicates that the system misses very few relevant items. For example, if a researcher searches for all occurrences of the verb "ሰመሄደ" (to go) in a scanned corpus and the system retrieves most documents containing morphological variants, then recall is high.

**Precision**

Precision measures how many of the retrieved documents are actually relevant. In Amharic scanned image retrieval systems, precision reflects how accurately the rules (e.g., shape-based matching, character profile recognition, or morphological pattern filters) identify correct document images without pulling in unrelated items.

Precision is defined as:

$$\text{Precision (P)} = \frac{\text{Number of Relevant Items Retrieved}}{\text{Total Number of Retrieved Items}}$$

High precision means that the system returns mostly relevant items with little "noise." For instance, when searching for the word "ኢትዮጵያ," if the system retrieves only scanned document that contain this word (and not those containing similarly-shaped but different terms), precision will be high.

**F-Measure (F-Score)**

While recall emphasizes completeness, precision emphasizes accuracy. Since both are important, the F-measure combines them into a single balanced score. For Amharic document retrieval, the F-measure helps determine how well the system handles trade-offs for example, whether strict morphological rules improve precision but reduce recall.

The standard definition is the harmonic mean of precision and recall:

$$\text{F} = \frac{2RP}{R+P}$$

The F-measure ranges from 0 to 1, where 1.0 represents perfect retrieval, meaning all relevant items are retrieved without including non-relevant ones. A higher F-measure indicates better overall system performance by balancing both precision and recall. In this way, it provides a single effectiveness score that reflects retrieval quality (Büttcher, Clarke, & Cormack, 2016).

In Amharic scanned document retrieval, precision, recall, and F-measure can be used to evaluate how effectively the rule-based NLP approach retrieves relevant document images, even without relying on machine learning-based methods.

## 3.2 Document Image Retrieval

Document image retrieval refers to the systematic process of locating and extracting relevant document images from a large and diverse repository of scanned or digitally stored documents.

Traditionally, this task was performed manually by librarians, archivists, legal researchers, and information specialists who relied on well-organized collections and dedicated indexing systems. With the rapid growth of digital archives, however, the need for efficient document image retrieval has become critical for automated information management systems in governmental, educational, historical, and research institutions.

Just as with text-based information retrieval, the foundation of an effective document image retrieval system lies in proper organization, preprocessing, and indexing. These steps ensure that retrieval is both fast and accurate. In languages such as Amharic, this need becomes even more pronounced. Amharic's unique syllabic script, complex character shapes, varying fonts, and common scanning imperfections make recognition and retrieval particularly challenging. For this reason, successful retrieval systems must be designed so that the information in the images can be accurately recognized, indexed, and made searchable.

In an OCR-based approach, the overall retrieval process begins with preprocessing, which enhances the scanned image quality and prepares it for reliable text extraction. Preprocessing operations may include de-skewing to correct tilted scans, noise removal to eliminate unwanted speckles and artifacts, binarization to convert the image into clearer foreground and background regions, and layout analysis to separate text from non-text elements such as images, borders, captions, or figures. Through careful preprocessing, the scanned page is transformed into a cleaner and more structured representation that allows the OCR engine to perform accurate character recognition.

After preprocessing, connected component analysis or segmentation techniques are often used to isolate individual textual elements such as words or lines. This segmentation helps the OCR engine by breaking down complex page layouts into manageable textual units. The OCR system then processes these segments to recognize characters and convert them into machine-readable text. OCR works by analyzing the patterns of light and dark regions in the image and matching them to known character shapes, allowing the scanned words to be converted into editable and searchable text. The recognized text extracted through OCR is used for feature and text extraction, where the meaningful textual content is retrieved and prepared for indexing. These extracted words, phrases, and structures form the basis of a searchable text database.

### 3.2.1 Document Image Processing Stages

Document image processing involves a series of well-organized operations that transform raw scanned images into clean, structured, and analyzable formats that are suitable for OCR, feature extraction, and retrieval. For an OCR-based Amharic document retrieval system, this stage is especially critical because the quality of image processing directly affects the accuracy of subsequent steps such as character recognition, linguistic normalization, and morphological analysis.

**Data Capturing**

The first stage in document image processing is data capturing, where physical paper documents are converted into digital form using devices such as scanners or digital cameras. During scanning, the device samples the surface of the document and represents it as a grid of small elements called pixels, where each pixel stores information about color or intensity. The type of pixel values depends on the image format: binary images represent each pixel as either 0 or 1, grayscale images use intensity values ranging from 0 to 255, and color images store three channels (red, green, and blue), each typically ranging from 0 to 255. This digitization process forms the foundation for further document analysis and retrieval tasks (Gonzalez & Woods, 2018).

**Binarization of Document Image**

Binarization is a key preprocessing step in document image processing that converts grayscale or color images into binary images. Its main purpose is to separate foreground text from the background, making the document easier to analyze. The quality of binarization significantly affects later stages such as segmentation and optical character recognition (OCR), since poor binarization can lead to broken or missing characters.

Several binarization techniques are commonly used, including global thresholding, Otsu's method, and adaptive thresholding approaches such as Sauvola's method. These methods perform differently depending on image conditions such as lighting variations, noise, and contrast. For degraded or unevenly illuminated documents such as historical Amharic manuscripts adaptive methods are often more effective because they adjust locally to image variations (Gonzalez & Woods, 2018; Sauvola & Pietikäinen, 2000).

**Noise Reduction**

Noise reduction is an important preprocessing step performed after binarization to remove unwanted distortions and artifacts in document images. These noises may originate from scanning devices, poor printing quality, ink bleeding, paper aging, or compression artifacts. In binary document images, a common type of noise is salt-and-pepper noise, where random black and white pixels appear across the image. This type of noise can negatively affect connected component analysis, reduce segmentation accuracy, and degrade the performance of OCR and subsequent text processing stages.

To mitigate these problems, several filtering techniques are commonly used, including median filtering, Gaussian filtering, and adaptive smoothing methods. Among these, median filtering is particularly effective for removing impulse noise while preserving important edges in document images. In more advanced document analysis systems, hybrid and adaptive filtering approaches are also used to better handle degraded or complex document conditions. Effective noise removal improves segmentation quality and enhances the reliability of feature extraction in later stages of document image processing (Bukhari, Shafait, & Breuel, 2016; Gonzalez & Woods, 2018).

**Skew Detection and Correction**

During scanning, photocopying, or digital imaging, document pages often become slightly tilted or misaligned, a problem known as skew. When text lines are not horizontally aligned, it negatively affects segmentation and character recognition, leading to reduced OCR accuracy. Even small rotation angles can distort text line structure and cause errors in word and line boundary detection, making skew correction an essential preprocessing step in document image processing systems.

Skew detection involves estimating the angle of rotation of a document image relative to its correct orientation. Once the skew angle is estimated, skew correction (deskewing) rotates the image back to a horizontal position, improving readability and enabling more accurate segmentation and recognition. Modern approaches for skew estimation include projection profile analysis, Hough transform-based methods, and machine-learning-assisted techniques, which are widely used in document analysis systems for their robustness under noisy and degraded conditions (Breuel, 2008; Rigaud et al., 2019; Gonzalez & Woods, 2018).

**Word Segmentation**

Word segmentation is the process of breaking a text line in a document image into individual word parts. It is an important step in document image retrieval systems because words are often used as the main units for organizing and searching information. In image-based retrieval, these word regions can also be treated as visual patterns, which allows the system to match and retrieve content directly from images without fully converting them into characters.

A common way to do word segmentation is by looking at the spacing between parts of a text line. This is often done using projection profiles, where the system detects gaps in pixel density. These gaps usually represent spaces between words, allowing the system to separate one word from another. This step is very important because if word segmentation is not accurate, it can affect later processes like feature extraction and recognition, which reduces overall retrieval performance. (Likforman-Sulem, Zahour, & Taconet, 2007)

**Normalization**

Normalization is an important step that comes after extracting word images from scanned documents using connected component analysis. At this stage, each word is detected and enclosed in a bounding box. However, these word images are often not uniform. They may differ in size, thickness, and scale because of variations in scanning quality, font styles, or handwriting differences.

If these features are compared directly, larger word images may unfairly influence the similarity calculation, while smaller ones may lose important detail. To avoid this problem, the feature values are rescaled into a common range, usually between 0 and 1. This makes sure that all word features are treated equally during comparison, regardless of their original size or shape. As a result, normalization improves the fairness and stability of the retrieval process by making feature comparison more consistent across different document images. (Pedregosa et al., 2011).

Feature normalization plays a critical role in an OCR-based retrieval pipeline because it directly influences how well the system can compare and match query text against the stored database. In the context of Amharic, the need for normalization is even more pronounced. The Amharic script uses syllabic characters, and slight differences in glyph forms often correspond to meaningful distinctions in the language. Without normalization, variations in image scale or stroke thickness could overwhelm these subtle linguistic cues, leading to inaccurate matching. By scaling features

such as aspect ratios, projection profiles, and shape descriptors into a unified range, the system ensures that each characteristic contributes proportionately to similarity computations.

### 3.2.2 Feature Extraction in OCR

Feature extraction is a fundamental step in Optical Character Recognition (OCR) and document image retrieval systems. It refers to the process of identifying and selecting meaningful visual patterns from document images so that text can be represented in a compact and discriminative form. Instead of analyzing the entire raw image, the system focuses on important characteristics such as edges, shapes, contours, and structural information of characters and words. This reduces computational cost and improves the efficiency of recognition and retrieval.

In modern document retrieval systems, extracted features are stored in a feature database and later used for comparison during search operations. When a user submits a query, the system extracts similar features and matches them against stored document representations to retrieve the most relevant results. The performance of the system strongly depends on how well these features capture the true structure of the text.

Feature extraction is particularly important for Amharic document images because the Ethiopic script contains complex character shapes and visually similar symbols. In addition, morphological variations in Amharic writing can change word forms significantly, making discriminative feature representation essential for accurate retrieval and recognition.

(Plötz & Roth, 2011).

Several types of features are commonly used in OCR and document image retrieval systems. These features help describe the shape, structure, and distribution of text within an image. Some of the commonly used features are explained below.

1. Width-to-Height Ratio: This feature describes the general shape of a word or character by comparing its width and height. It helps distinguish long words from short or compact ones.
2. Word Area Density Ratio: This measures how dense a word image is by calculating the proportion of black pixels within the word area.
3. Center of Gravity: This feature shows how pixels are distributed inside a word image and helps identify differences in character arrangement and structure.
4. Vertical Projection Profile: This is obtained by summing pixel values along each column of the image. It captures vertical writing patterns and structural information.

5. Horizontal Projection Profile: This feature sums the foreground pixels across each row of the image and helps represent horizontal text patterns.
6. Upper Word Profile: This measures the distance from the upper boundary of the image to the first black pixel in each column, helping capture the upper shape of the word.
7. Lower Word Profile: Similar to the upper profile, this measures the distance from the lower boundary to the first foreground pixel, representing the lower contour of the word.
8. Top Word Shape: This feature captures the shape of the upper strokes of characters and helps differentiate similar-looking characters.
9. Bottom Word Shape: This represents the lower stroke patterns of characters and provides additional structural information for recognition.

Generally, feature extraction helps represent document images in a simpler and more meaningful way. In Amharic OCR systems, it plays a major role in improving character recognition and increasing the accuracy of document retrieval from scanned document collections.

### 3.2.4 Similarity Measures

In large document collections, a single query often retrieves many partially relevant documents. Therefore, retrieval systems must not only find matching documents but also rank them based on relevance. This is achieved using similarity measures, which calculate how closely a document matches a query.

Modern information retrieval systems mainly use two approaches:

1.Semantic-based similarity: This method focuses on understanding the meaning of words and sentences using techniques like word embeddings and transformer models such as BERT. It is more accurate, especially for morphologically rich languages like Amharic, but it requires high computational resources and is less efficient for large-scale or real-time systems.

2. Statistical or feature-based similarity: This traditional method represents documents using statistical features such as TF-IDF and cosine similarity. It does not understand meaning directly but relies on term frequency patterns. Because of its simplicity and efficiency, it is still widely used in practical IR systems (Singhal, 2001; Manning et al., 2008).

These findings suggest that combining traditional statistical methods with modern semantic techniques gives better results for Ethiopian languages.

**Some widely used similarity techniques include:**

- Euclidean distance: Measures the straight-line distance between feature vectors of the query and document word.
- Cosine similarity: Computes the cosine of the angle between feature vectors, useful for assessing relative orientation of features.
- Correlation-based measures: Evaluate how feature patterns in the query correlate with those in document words.
- By applying these similarity measures, the retrieval system can efficiently rank and return the most relevant Amharic document images to the user, ensuring that the top results are both accurate and meaningful.

**Euclidean Distance**

Euclidean distance is a simple and widely used similarity measure that comes from the Pythagorean theorem. It calculates the straight-line distance between two points represented as feature vectors in a multi-dimensional space. In information retrieval or image processing, each document or image is converted into numerical features, and similarity is determined by how close these feature vectors are to each other. A smaller Euclidean distance means the two items are more similar, while a larger distance means they are more different. Because of its simplicity and effectiveness, Euclidean distance is still widely used in machine learning and data analysis for comparing feature-based representations (Gallet & Gowanlock, 2022).

$$\left\|\vec{D}-\vec{Q}\right\|=\sqrt{\sum_{i=1}^{p}(xi-yi)^{2}}.$$

Here, $x_i$and $y_i$are the ith feature values of the query image $Q$and the database image $D$, respectively. In Amharic document retrieval, this is particularly useful for comparing geometric and structural features of word images, such as vertical and horizontal projections, profiles, and shape ratios. For example, two images of the Amharic word "መምህር" with slightly different sizes or rotations can be compared effectively using Euclidean distance on their normalized feature vectors.

**Normalized Cross-Correlation (NCC)**

Normalized Cross-Correlation (NCC) is a statistical similarity measure used in image processing to compare two images by measuring how strongly their corresponding pixel or feature values

match. It is especially useful in image matching and registration because it is less sensitive to changes in brightness and contrast. A higher NCC value indicates stronger similarity between the images.

NCC is widely applied in image matching, template matching, and registration tasks in modern computer vision systems (Morelli et al., 2022). It is particularly effective in matching word images when local variations in the image, such as skew or minor noise, may occur. Given two sets of variables, $X = \{x_1, x_2, \dots, x_n\}$and $Y = \{y_1, y_2, \dots, y_n\}$, the NCC is calculated as:

$$R(x,y)=\frac{\sum_i (xi - \bar{X})(yi - \bar{y})}{\sqrt{\sum_i (xi - \bar{x})2}\sqrt{\sum_i (yi - \bar{y})2}}$$

Where $\bar{X}$and $\bar{Y}$are the mean values of $X$and $Y$. The resulting $R$ranges from -1 to 1, where 1 indicates perfect similarity, 0 indicates no correlation, and -1 indicates inverse correlation. For Amharic word images, NCC can help in matching words that are visually similar but may have slight distortions due to scanning or handwriting differences.

**Cosine Distance**

Cosine distance is a similarity measure that compares two vectors by calculating the cosine of the angle between them, rather than their magnitude. In document image retrieval, each image or text sample is represented as a feature vector, and similarity is determined by how closely the directions of these vectors align. This makes cosine distance especially useful when the size of documents varies but their content similarity is important. So, it is computed as:

$$\text{Sim}(\vec{q}, \vec{d}) = \frac{\sum_{i=1}^{n} qi.di}{\sqrt{\sum_{i=1}^{n} qi2 \sum_{i=1}^{n} di2}}.$$

.

Here, $q_i$and $d_i$are the components of the vectors. Cosine similarity ranges from 0 (completely independent) to 1 (exactly the same). In Amharic image retrieval, cosine similarity is advantageous because it is less sensitive to variations in word size and scaling, focusing on the pattern of features rather than absolute values.

## 3.3 Related Works

Research in the area of document image retrieval, especially for low-resource languages such as Amharic, is still relatively limited. While extensive studies have been carried out for Latin-based scripts and other widely used languages, few efforts have focused on retrieving Amharic document images without relying on Optical Character Recognition (OCR). In this section, I discuss the key studies that form the foundation of my research, highlighting the methods they used, the features they focused on, and the techniques applied for matching and retrieving document images.

Word image retrieval focuses on identifying and matching word images without relying on OCR. In this approach, images are represented using distinctive structural and geometric features such as word height, width, projection profiles, and stroke density. These features help distinguish visually similar word images and support efficient matching in large document collections. Such feature-based methods are widely used in content-based image retrieval systems, where retrieval is performed based on visual similarity rather than text recognition (Alzoubi et al., 2024).

Document image retrieval systems often use projection-based features to represent word images. In this approach, horizontal and vertical projection profiles are extracted to capture the overall shape and structure of words. These features are useful for matching word images across different documents, even when noise, scanning distortions, or font variations are present. Feature normalization and appropriate similarity measures are important for improving retrieval accuracy in such systems (Balasubramanian et al., 2006).

Word image retrieval focuses on representing word images using multiple structural and statistical features such as top and bottom profiles, word area density, and centroid-based features. Combining multiple features helps create a more robust representation of word shapes, especially in multi-font and multi-size document images. These feature vectors are then compared using similarity measures such as Euclidean distance or cosine similarity to rank retrieved results based on relevance. This multi-feature approach improves the robustness of OCR-free document image retrieval systems (Zagoris et al., 2021; Balasubramanian et al., 2006).

Amharic document image retrieval searches scanned documents without full OCR by using visual features from word images. Mesfin Worku (2009) proposed a recognition-free approach that represents words using structural features and compares them using similarity measures like Euclidean distance and cosine similarity. However, the method relies mainly on visual and

statistical features and does not fully capture Amharic's rich morphological variations, which can reduce retrieval accuracy. This shows the importance of adding morphological processing to improve performance in Amharic retrieval systems (Worku, 2009).

### 3.4 English and Other Non-Amharic Document Images Retrieval

Research on document image retrieval (DIR) for English and other non-Amharic languages has laid a strong foundation for word-based image retrieval systems, particularly in scenarios where OCR is not employed. (Anand's 2007). Anand et al. highlighted the challenge of direct image matching, which is computationally expensive and impractical for large-scale collections. To overcome this, they proposed a system based on locality-sensitive hashing (LSH), where word images are first segmented automatically, features are extracted at the word level, and then hashed into multiple tables using content-sensitive hash functions. This hashing strategy increases the likelihood that visually similar words fall into the same hash bucket, allowing sub-linear time retrieval without sacrificing accuracy. Experiments on Indian language books demonstrated that their approach could retrieve word images efficiently, achieving approximately 90% recall and precision. The system uses a combination of scalar, profile, structural, and transform-domain features, demonstrating that carefully designed feature extraction and hashing can significantly enhance document image retrieval performance.

Document image retrieval systems for digital libraries often avoid full OCR by using word image matching techniques. In this approach, word images are extracted offline and represented using compact feature codes or feature vectors that capture their visual structure. These representations are stored in an index for efficient retrieval. When a user submits a text query, it is converted into the same feature representation, and matching is performed using approximate string or feature similarity techniques. Documents are then ranked based on relevance measures such as term frequency or feature similarity, enabling efficient retrieval from large scanned document collections (Balasubramanian et al., 2006).

Word shape coding approaches for document image retrieval aim to search scanned documents without using OCR. Lu et al. (2008) proposed a method that converts word images into sequences of shape-based codes using structural features such as ascenders, descenders, holes, and reservoirs. These word representations are generated after preprocessing steps such as binarization, smoothing, and word segmentation using projection profiles. The resulting codes form a compact

representation of each word image, which can be efficiently compared using sequence matching or similarity measures such as edit distance. This approach enables fast and robust retrieval of document images, even under variations in font style, noise, and document degradation (Lu et al., 2008).

(Marinai et al., 2007) developed a recognition-free DIR system for digital libraries, allowing retrieval based on layout and textual queries. The system integrates font-independent word indexing with layout-based retrieval, using a three-component framework consisting of indexing, query formulation, and similarity computation. Pages are processed with layout analysis tools to extract homogeneous regions, and words are encoded using character labels. Queries can be textual or layout-based, and documents are ranked according to similarity in both text and layout. This approach emphasizes the combination of textual and structural features for more accurate document retrieval in digital archives.

(Zagoris et al., 2010) proposed a word spotting system for document image archives that bypasses character recognition. The system represents both query words and document words using seven robust features, including width-to-height ratio, word area density, center of gravity, vertical and horizontal projections, and upper/lower grid features. Preprocessing steps such as noise removal, binarization, and word segmentation (via connected component analysis) ensure accurate feature extraction. Queries are processed identically to the documents, and matching is performed through feature vector comparison. Experiments reported mean precision of 87.8% and mean recall of 99.26%, highlighting the effectiveness of word feature-based retrieval techniques, even in noisy and multi-font document images.

### 3.4.1 Amharic Document Images Retrieval

Research on Amharic document image retrieval (DIR) is still relatively limited compared to English and other widely-studied languages. However, several studies have explored techniques to retrieve Amharic document images efficiently, often without relying on OCR, by using word-level features and similarity measures.

Jawahar, Balasubramanian, and colleagues proposed document image retrieval methods that enable searching scanned document databases without relying on OCR. Their approach represents word images using multiple structural and statistical features, such as projection profiles, pixel density, ink-to-background ratio, and statistical moments. These features are used to describe word

shapes in a compact form for indexing and retrieval. Similarity between word images is computed using distance-based methods, and ranking is performed based on query relevance. This feature-based retrieval framework demonstrated that OCR-free indexing can effectively support multilingual document image retrieval, including languages such as Amharic, English, and Hindi (Balasubramanian et al., 2006).

Abreham (2010) proposed an Amharic document image retrieval system that improves search efficiency in scanned document collections without using OCR. The system introduces an inverted index structure to organize word image features and remove non-discriminative terms using filtering techniques such as inverse document frequency. Similarity between query and document word images is measured using cosine similarity, and affix removal is applied to reduce words to their root forms. The results showed improved retrieval efficiency and performance gains, demonstrating the effectiveness of indexing-based approaches for Amharic document image retrieval systems (Abreham, 2010).

Despite their usefulness, these methods still have some important limitations. They mainly depend on visual similarity and do not fully consider the linguistic richness of the Amharic language. Because of this, they often have difficulty handling variations in word forms. In addition, they are not very effective when dealing with complex user queries and may struggle to perform well on large collections of documents.

In my study, a different approach is taken by integrating NLP techniques with document image processing. Instead of relying only on similarity measures, the proposed system uses linguistic rules to normalize Amharic words and improve indexing and matching. In this way, the work not only addresses the limitations of existing methods but also provides a more accurate and language-aware retrieval system.

# CHAPTER FOUR

## PROPOSED SYSTEM ARCHITECTURE

### 4.1 proposed system and experimental results for Amharic document retrieval

This section presents the proposed Amharic Document Image Retrieval System (ADIRS) and the experimental results used to evaluate its performance. The system is designed to retrieve relevant scanned Amharic documents by integrating Optical Character Recognition (OCR) with Natural Language Processing (NLP) techniques. Unlike traditional document image retrieval approaches that mainly depend on visual word-image matching, the proposed system converts scanned document images into editable text and performs retrieval using textual and linguistic analysis. This approach improves retrieval effectiveness by addressing the morphological complexity and orthographic variations of the Amharic language.

As illustrated in Figure 4.1, the proposed system consists of several interconnected modules that transform scanned document images into searchable textual representations. The process begins with image preprocessing operations such as noise removal and binarization to improve OCR accuracy. The OCR module then converts scanned Amharic document images into machine-readable text. After text extraction, the system applies NLP techniques including orthographic normalization, stop-word removal, stemming, and correction of common OCR-related errors. These processes reduce word variation and improve indexing consistency.

The system applies morphological analysis to normalize different surface forms of Amharic words into common root representations. This allows the retrieval process to identify relevant documents even when words appear in different grammatical or inflectional forms. The processed text is then indexed to support efficient document retrieval.

Users interact with the system through a graphical user interface by entering Amharic text queries. The query terms undergo the same preprocessing and normalization procedures applied during document indexing to maintain consistency between query and document representations. The processed queries are matched against the indexed corpus using text-based similarity measures, and retrieved documents are ranked according to their relevance scores.

The proposed Amharic document image retrieval with the approach of NLP consists of four major components:

1. User Query Processing Module: accepts user queries and applies normalization, stop-word removal, and stemming.
2. NLP stage and Index Construction Module: performs linguistic processing and constructs the searchable text index.
3. Document Image Processing and OCR Module: preprocesses scanned document images and extracts editable text through OCR.
4. Document Retrieval and Ranking Module: computes similarity scores, ranks retrieved documents, and returns relevant results.

To evaluate the effectiveness of the proposed system, experiments were conducted using scanned Amharic document collections. System performance was evaluated using standard information retrieval metrics such as precision, recall, F-measure, and retrieval time. The experimental results demonstrate that integrating OCR with rule-based Amharic NLP techniques improves retrieval accuracy and efficiency for scanned Amharic document collections.

In this architecture, all computationally intensive tasks such as document image preprocessing, OCR conversion, linguistic normalization, and index construction are performed offline to ensure fast and efficient system response during search. The online component is limited to query processing, linguistic normalization of query terms, index lookup, and result ranking. This clear separation between offline and online processing optimizes system performance and enables near real-time retrieval, even when working with large collections of scanned Amharic documents. The subsequent sections of this chapter provide detailed descriptions of each component, outlining their internal modules, processing workflows, and the algorithms that support the overall operation of the proposed system.

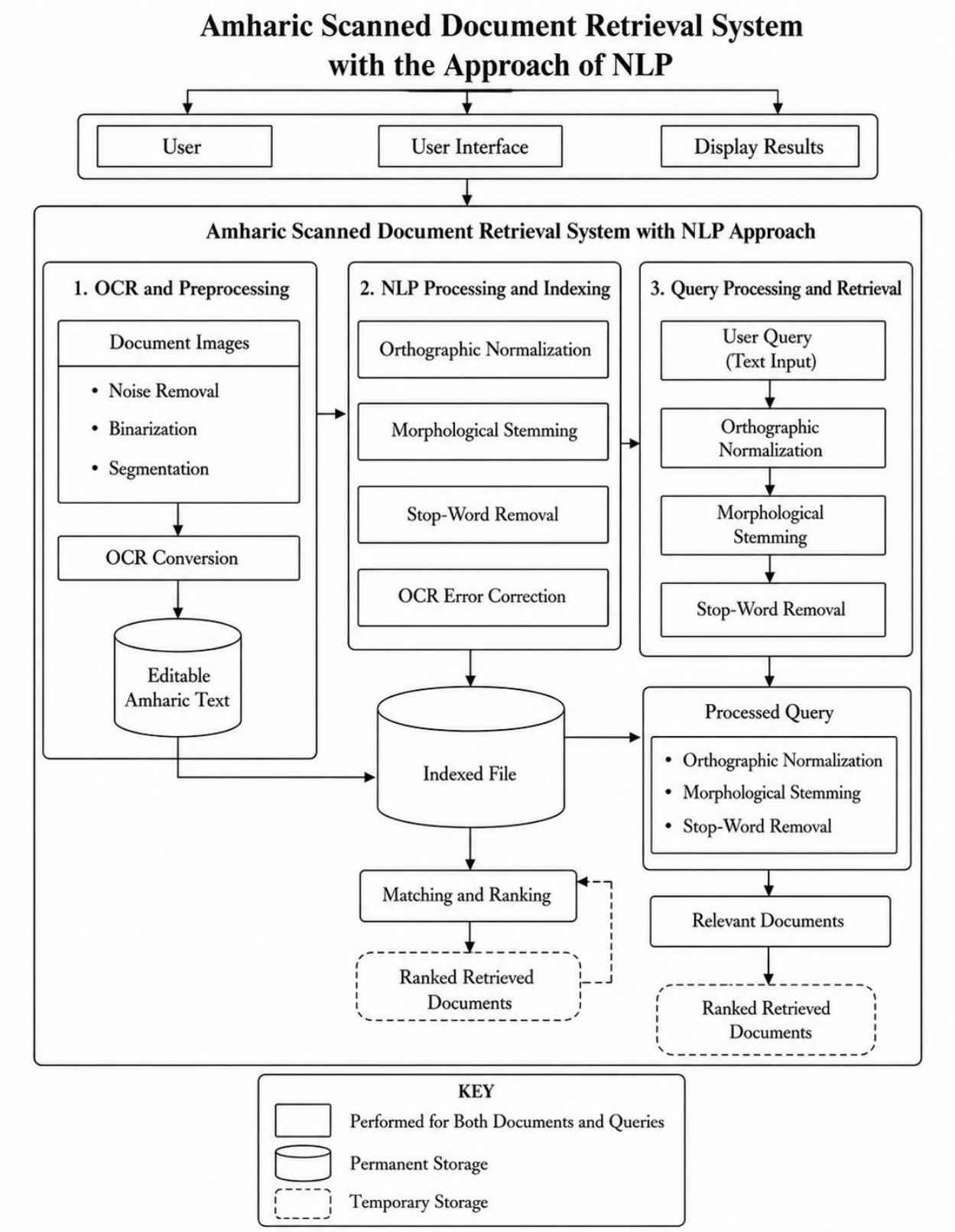



Figure 4.1 proposed system architecture

## 4.2 Amharic Word Synthesizer and Textual Feature Corpus Preparation

Amharic is a morphologically rich language in which a single root word can generate a large number of surface forms through inflectional and derivational processes. These processes include prefixation, suffixation, infixation, circumfixing, and complex concatenative patterns that significantly increase lexical variation in real-world texts. Such morphological richness presents a serious challenge for scanned document retrieval systems that rely on OCR output, as different surface forms of the same root may be treated as unrelated terms during indexing and search. Without linguistic normalization, this variation leads to reduced recall and fragmented retrieval results.

To address this challenge, the proposed system integrates Amharic morphological processing component that captures the rich and highly productive structure of the language. In Amharic, a single root can appear in many different surface forms due to verb conjugation, derivation, pluralization, gender marking, and possessive constructions, which often creates serious vocabulary mismatch problems in information retrieval systems. To handle this, the system applies a rule-based Amharic Word Synthesizer that generates valid morphological variants from a given root word, improving the alignment between user queries and document content while reducing lexical sparsity. Prior research has shown that explicit morphological modeling significantly improves Amharic NLP tasks, especially in low-resource settings where data sparsity is a major challenge (Gasser, 2011; Mekonnen et al., 2021).

Unlike earlier Amharic Document Image Retrieval (ADIR) systems that relied on visual word features, the proposed system uses the synthesizer to operate entirely at the textual level after OCR conversion. The OCR-extracted text from scanned documents is processed using rule-based normalization techniques, where surface word forms are mapped to their corresponding root representations using synthesizer-generated rules. This process ensures that morphologically related words are consistently grouped under a unified lexical representation during indexing.

To construct the textual feature corpus, all synthesized surface forms are processed through a normalization pipeline that includes orthographic standardization, stop-word removal, and morphological root mapping. Each normalized term is then stored in the index along with its document identifiers and frequency statistics. The resulting corpus is therefore not a collection of

visual word features, but rather a linguistically enriched textual index that captures the full morphological space of Amharic root words. This textual feature corpus forms the backbone of the retrieval system and enables efficient matching between user queries and OCR-derived document content.

Amharic information retrieval systems face challenges because the language has a rich and complex morphological structure, where a single root word can appear in many different surface forms through inflection and derivation. This variation makes it difficult for traditional surface-level matching methods to correctly identify words that share the same meaning, which often reduces retrieval accuracy. To handle this problem, morphological analysis and root-based representation are commonly used in Amharic NLP and information retrieval systems, as they help normalize different word forms into a single unified representation. Studies have shown that applying morphological processing in Amharic text retrieval significantly improves how documents are represented and increases overall retrieval performance, especially in low-resource language settings where word variation strongly affects matching accuracy (Yeshambel, Mothe, & Assabie, 2022).

The use of this synthesizer eliminates the need to rely solely on exact surface-form matching during retrieval. Even when a user submits a query using an inflected or derived word form that does not appear verbatim in a document, the system can still retrieve relevant documents by matching normalized root representations. This design choice ensures robust and linguistically informed retrieval, fully aligning the system with the objectives of OCR-based, rule-driven Amharic scanned document retrieval.

**Research Investigation and Identified Gap**

An extensive review of previous studies on Amharic OCR, Natural Language Processing (NLP), and Information Retrieval (IR) was conducted to understand the current research landscape. The investigation revealed that most existing studies focused on individual tasks such as text extraction, language processing, or document retrieval. OCR-based research mainly concentrated on converting scanned documents into machine-readable text, while NLP studies focused on processing Amharic text through tokenization, normalization, and stop-word removal. Similarly, information retrieval studies were primarily designed for digital text documents rather than scanned document images. As a result, there is still a lack of an integrated system that can

effectively process, index, and retrieve information from scanned Amharic documents. This gap motivated the development of the proposed research.

## 4.3 Document Image processing

Document image processing refers to the set of techniques and procedures used to analyze, enhance, and prepare scanned document images for information extraction and retrieval. In document image retrieval systems, this process plays a fundamental role in improving the quality of scanned documents before Optical Character Recognition (OCR) and Natural Language Processing (NLP) operations are performed. The main objective of document image processing is to convert raw scanned documents into a clean and structured form that can be accurately recognized, indexed, and retrieved by the system.

Traditionally, document retrieval tasks were performed manually by librarians, archivists, and information specialists who relied on physical filing systems and cataloging methods to locate documents. However, with the rapid growth of digital repositories and scanned document collections, manual approaches became inefficient, time-consuming, and difficult to manage. As a result, automated document image processing and retrieval systems have become increasingly important in educational institutions, libraries, government offices, legal organizations, and historical archives.

In OCR-based document retrieval systems, scanned document images are first transformed into machine-readable textual information using Optical Character Recognition technology. The effectiveness of OCR-based retrieval systems highly depends on the quality of document image processing performed before recognition. This requirement is especially important for the Amharic language because Amharic uses a complex Ethiopic script with large character sets, syllabic structures, visually similar characters, multiple font styles, and rich morphological variations. In addition, scanned Amharic documents may contain noise, skewness, faded text, broken characters, and scanning distortions, which can negatively affect OCR accuracy and retrieval performance if not properly handled.

Therefore, document image preprocessing is considered a mandatory stage in the proposed System. The preprocessing stage improves image quality and prepares document images for accurate segmentation and OCR recognition. Several preprocessing operations are applied, including grayscale conversion, noise removal, binarization, skew detection and correction, image

normalization, and layout analysis. Noise removal techniques help eliminate unwanted scanning artifacts and background distortions, while binarization converts grayscale images into binary representations to simplify text extraction. Skew correction aligns tilted document images into a proper orientation, thereby improving segmentation and recognition accuracy (Yang et al., 2024). After preprocessing, the document image undergoes segmentation. Segmentation divides the document into smaller meaningful components such as text lines, words, or characters depending on the system design. In the proposed system, line segmentation and word segmentation are performed using projection-based and connected component analysis techniques. These techniques help isolate individual Amharic textual units from the scanned document image. Accurate segmentation is essential because OCR performance directly depends on the quality of segmented text regions.

The segmented word images are then passed to the OCR module, where Amharic characters and words are recognized and converted into editable text form. Tesseract OCR with Amharic language support is employed in this study to perform text recognition. Once textual information is extracted, the recognized text is further processed using rule-based NLP techniques including tokenization, stop-word removal, normalization, and stemming. These operations help reduce morphological variations and improve retrieval consistency.

Finally, the processed textual features are indexed and stored in the retrieval database. When a user submits an Amharic query, the system compares the query terms with indexed document features using TF-IDF ranking and hybrid similarity matching techniques to retrieve the most

relevant scanned document images

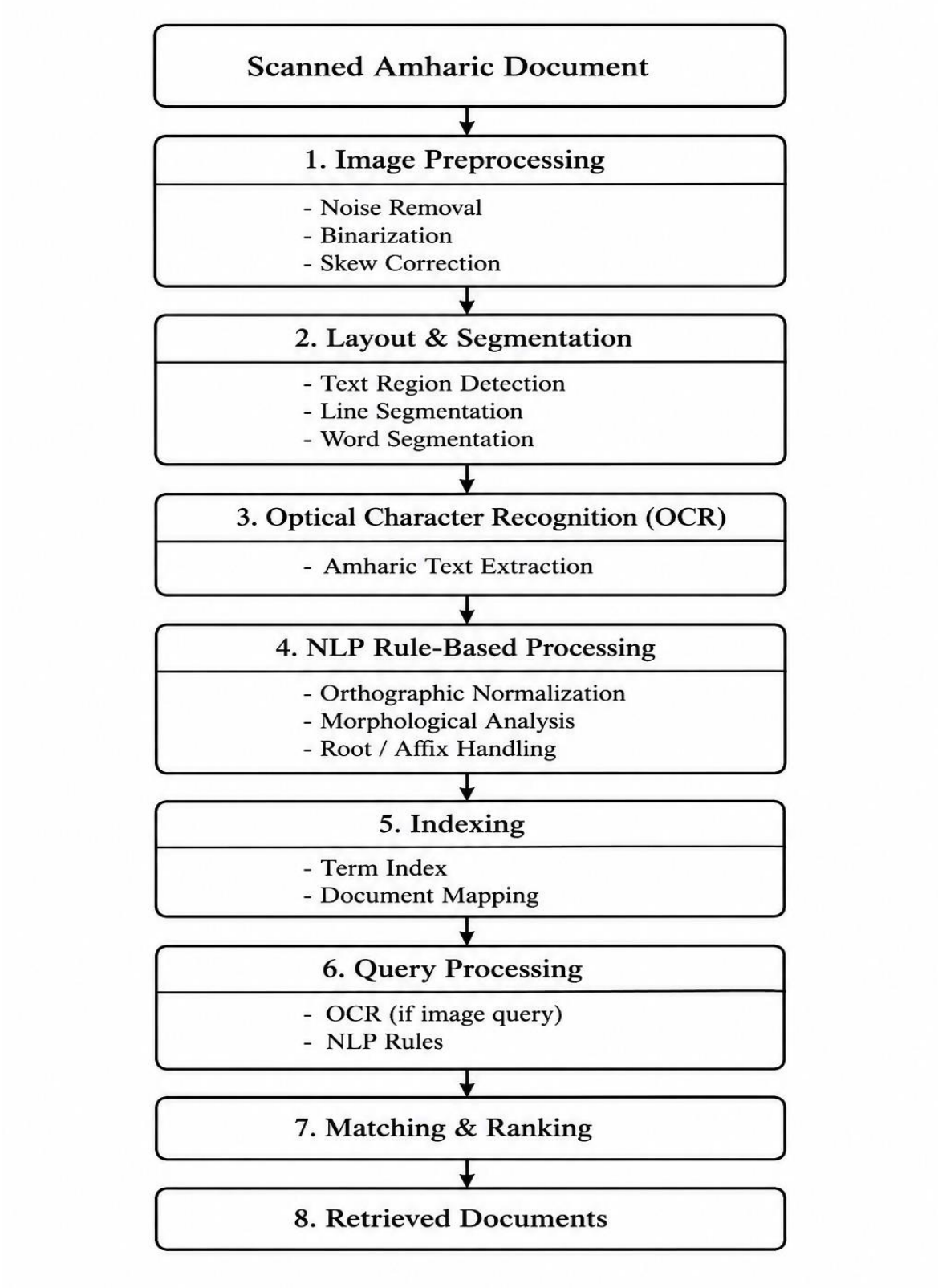



Figure 4.2 document image processing

### 4.3.1 Noise Removal and experiment

Document noise refers to unwanted variations in pixel intensity that do not represent actual written text. In scanned documents, this kind of noise usually comes from the scanning device, paper texture, dust particles, uneven lighting, or compression during saving. In OCR systems, such distortions make it harder to correctly detect and separate characters because they break or blur character boundaries, which reduces the accuracy of feature extraction and recognition (Gonzalez & Woods, 2018).

To address this problem, the proposed system uses a median filtering technique during preprocessing in Python. This method works by replacing each pixel with the median value of its surrounding neighbors, which makes it very effective at removing random noise such as salt-and-pepper artifacts while keeping important edges intact. This is especially important for Amharic text, where small changes in strokes can affect meaning and character identity. By choosing an appropriate kernel size through experimentation, the system is able to reduce noise without losing important text details, leading to cleaner images and better OCR performance overall (Gonzalez & Woods, 2018).

```
import cv2

import numpy as np
from tkinter import filedialog
from tkinter import Tk

def apply_noise_removal(image_path, kernel_size=3):
    # Read as grayscale
    img = cv2.imread(image_path, cv2.IMREAD_GRAYSCALE)
    if img is None:
        print("Error: Could not load image.")
        return None, None

    # Apply median filter (noise removal)
    denoised = cv2.medianBlur(img, kernel_size)

    return img, denoised

def main():
    # Open file dialog to select image
    root = Tk()
    root.withdraw()
    file_path = filedialog.askopenfilename(
```

```
        title="Select a scanned image with noise",
        filetypes=[("Image files", "*.png *.jpg *.jpeg *.bmp *.tiff")]
    )
    if not file_path:
        print("No file selected.")
        return

    print(f"Processing: {file_path}")
    original, denoised = apply_noise_removal(file_path, kernel_size=3)
    if original is None:
        return

    # Save result
    output_path = "noise_removed_output.png"
    cv2.imwrite(output_path, denoised)
    print(f"Denoised image saved as: {output_path}")

    # Optional: Display side-by-side
    cv2.imshow("Original (with noise)", original)
    cv2.imshow("After Median Filter (noise removed)", denoised)
    cv2.waitKey(0)
    cv2.destroyAllWindows()

if __name__ == "__main__":
    main()
```

sample implementation source code for noise removal

### 4.3.2 Binarization and experiment

Binarization is the next important step after noise removal in the preprocessing pipeline. It involves converting a grayscale image into a binary form, where each pixel is classified as either foreground (text) or background. This step is essential in OCR systems because recognition algorithms perform more effectively when text is clearly separated from the background. By increasing contrast and simplifying the image structure, binarization improves segmentation accuracy and overall recognition performance (Gonzalez & Woods, 2018).

In this research, binarization is implemented using a thresholding approach, where each pixel is assigned, a value based on whether its intensity is above or below a selected threshold. A widely used and reliable method is Otsu's thresholding technique, which automatically determines an optimal threshold by minimizing the variance within foreground and background classes (Otsu,

1979). Applying binarization after noise removal produces a cleaner and more structured binary image, making character boundaries clearer and reducing segmentation errors. This step ultimately strengthens the OCR process and supports more accurate text extraction for the subsequent NLP-based retrieval system (Gonzalez & Woods, 2018).

```
import cv2
import numpy as np
from tkinter import filedialog
from tkinter import Tk

def binarize_image(image_path):
    # Read as grayscale
    img = cv2.imread(image_path, cv2.IMREAD_GRAYSCALE)
    if img is None:
        print("Error: Could not load image.")
        return None

    # Noise removal (median filter)
    denoised = cv2.medianBlur(img, 3)

    # Otsu binarization
    _, binary = cv2.threshold(denoised, 0, 255, cv2.THRESH_BINARY +
cv2.THRESH_OTSU)

    # Ensure text is black on white background
    if np.mean(binary) > 127:
        binary = cv2.bitwise_not(binary)

    return binary

def main():
    # Open file dialog to select image
    root = Tk()
    root.withdraw()  # Hide the main window
    file_path = filedialog.askopenfilename(
        title="Select a scanned image",
        filetypes=[("Image files", "*.png *.jpg *.jpeg *.bmp *.tiff")]
    )
    if not file_path:
        print("No file selected.")
        return

    print(f"Processing: {file_path}")
```

```
    binary = binarize_image(file_path)
    if binary is None:
        return

    # Save result
    output_path = "binarized_output.png"
    cv2.imwrite(output_path, binary)
    print(f"Binarized image saved as: {output_path}")

    # Optional: Display the result (comment out if no GUI support)
    cv2.imshow("Original", cv2.imread(file_path, cv2.IMREAD_GRAYSCALE))
    cv2.imshow("Binarized", binary)
    cv2.waitKey(0)
    cv2.destroyAllWindows()

if __name__ == "__main__":
    main()
```

Sample implementation for Binarization

### 4.3.3 Line segmentation and experiment

Line segmentation is the process of breaking a scanned document into individual lines of text before Optical Character Recognition (OCR) is carried out. This step is important because it helps the system organize the document into clear reading units, making it easier to process and interpret the text accurately. By separating each line, the system reduces confusion caused by overlapping or closely packed text, which ultimately improves OCR performance in document analysis tasks (Smith, 2007).

In this study, line segmentation is applied after preprocessing to divide Amharic documents into separate lines. This is particularly important for Ethiopic script, where characters are densely written and closely spaced, making layout analysis more challenging. Without proper segmentation, the OCR system may mistakenly mix words or characters from neighboring lines, leading to recognition errors. The system detects line boundaries by analyzing horizontal pixel distribution and identifying gaps between text regions. After segmentation, each line is processed individually for OCR and further NLP-based retrieval, resulting in more accurate and well-structured text extraction.

```
import cv2
import numpy as np
from tkinter import filedialog
from tkinter import Tk
```

```
def segment_lines(binary_image):
    # Sum black pixels horizontally (assuming text is black, background white)
    horizontal_proj = np.sum(binary_image == 0, axis=1)

    lines = []
    in_line = False
    start = 0
    for i, val in enumerate(horizontal_proj):
        if val > 0 and not in_line:
            in_line = True
            start = i
        elif val == 0 and in_line:
            in_line = False
            lines.append((start, i))
    return lines

def main():
    root = Tk()
    root.withdraw()
    file_path = filedialog.askopenfilename(title="Select binarized image",
filetypes=[("Images", "*.png *.jpg")])
    if not file_path:
        print("No file selected.")
        return

    # Load binary image
    img = cv2.imread(file_path, cv2.IMREAD_GRAYSCALE)
    if img is None:
        print("Error loading image.")
        return

    # Ensure binary (0=black, 255=white)
    _, binary = cv2.threshold(img, 127, 255, cv2.THRESH_BINARY)
    # Invert if needed so text is black
    if np.mean(binary) < 127:
        binary = cv2.bitwise_not(binary)

    # Detect line boundaries
    lines = segment_lines(binary)

    # Draw rectangles around each detected line on a colour copy
    color_img = cv2.cvtColor(binary, cv2.COLOR_GRAY2BGR)
    for (start, end) in lines:
```

```
        cv2.rectangle(color_img, (0, start), (binary.shape[1], end), (0, 255, 0),
2)

    output_path = "line_segmentation_result.png"
    cv2.imwrite(output_path, color_img)
    print(f"Saved result to {output_path}")
    cv2.imshow("Line Segmentation", color_img)
    cv2.waitKey(0)
    cv2.destroyAllWindows()

if __name__ == "__main__":
    main()
```

Sample implementation source code for Line segmentation

### 4.3.4 Word Segmentation and experiment

Word segmentation is the process of dividing each line of text into individual words after line segmentation has been completed. In Amharic and other Ethiopic scripts, this step is especially important since words are often written closely together, and inconsistent spacing can easily lead to recognition errors if not handled properly (Abdurahman et al., 2021).

In the proposed system, word segmentation is applied after line segmentation on scanned Amharic documents. The system detects word boundaries by analyzing gaps and connected components within each line image. Because Amharic is a morphologically rich language, accurate segmentation is essential to ensure that words are properly separated before further processing. Recent research also highlights that proper segmentation improves the performance of downstream NLP tasks such as recognition and text processing (Feyisa & Hailu, 2023). After segmentation, each word is processed individually in the OCR pipeline, which improves overall accuracy and reduces recognition mistakes.

```
import cv2
import numpy as np
from tkinter import filedialog
from tkinter import Tk

def segment_words(line_image):
    # Ensure binary (0=black text, 255=white background)
    _, binary = cv2.threshold(line_image, 127, 255, cv2.THRESH_BINARY)
    if np.mean(binary) < 127:
        binary = cv2.bitwise_not(binary)
```

```
    # Vertical projection: count black pixels in each column
    vertical_proj = np.sum(binary == 0, axis=0)

    words = []
    in_word = False
    start = 0
    for i, val in enumerate(vertical_proj):
        if val > 0 and not in_word:
            in_word = True
            start = i
        elif val == 0 and in_word:
            in_word = False
            words.append((start, i))
    return words

def main():
    root = Tk()
    root.withdraw()
    file_path = filedialog.askopenfilename(title="Select a text line image
(binarised)", filetypes=[("Images", "*.png")])
    if not file_path:
        print("No file selected.")
        return

    line_img = cv2.imread(file_path, cv2.IMREAD_GRAYSCALE)
    if line_img is None:
        print("Error loading image.")
        return

    words = segment_words(line_img)
    color_img = cv2.cvtColor(line_img, cv2.COLOR_GRAY2BGR)
    for (start, end) in words:
        cv2.rectangle(color_img, (start, 0), (end, line_img.shape[0]), (0, 255,
0), 1)

    cv2.imshow("Word Segmentation", color_img)
    cv2.waitKey(0)
    cv2.destroyAllWindows()

if __name__ == "__main__":
    main()
```

Sample implementation source code for word segmentation

## 4.4 NLP Processing

### Orthographic Normalization

Orthographic normalization cleans up common OCR errors by mapping visually similar Amharic characters to a single standard form. The rules are simple: any occurrence of ሐ, ሓ, or ኀ is replaced with ሀ; ሠ is replaced with ሰ; and ፀ is replaced with ጸ. This substitution is applied to every character in the OCR output, unifying variant spellings that arise from scan noise or font differences. For example, "ሐምሌ" becomes "ሀምሌ", ensuring that the same word is not split into multiple index entries. This step is fast, effective, and recovers many recognition errors at almost no computational cost.

### Morphological Analysis

Morphological analysis, or stemming, reduces Amharic words to their core stems by stripping off a comprehensive set of known affixes in a fixed order. The stemmer first checks for circumfixes the pairs አ…ነ and ተ…ነ – and removes both parts when found. Next, it removes any of a long list of prefixes, always trying the longest match first; these include subject markers (ይ, ት, እ, ን), causative markers (አስ, አሳ, አሰ, አስተ), passive and reflexive markers (ተ, እተ, እንተ), nominalizers (ማ, መ, ም, ሚ, ሙ), relative forms (ያስ, ያሳ, ያሰ, ያስተ, የተ, የታ), and complex relative–causative combinations (የማ, የመ, የም, የሚ, የሙ). After prefixes, the stemmer removes a similarly extensive list of suffixes, again longest first; these include tense and aspect markers (ኩ, ክ, ሽ, ን, ችሁ, ኡ, ኧ, ኣቸው, ኣት, etc.), subject-verb agreement suffixes (ሁ, ህ, ሽ, ን, ችሁ, ኡ, ኤ, ኤች), complex verb endings (ኛለሁ, ኛለህ, ኛለሽ, ኛል, ኛለች, ኛለን, ኛላችሁ, ኛሉ), past auxiliary forms (ነበርኩ, ነበርክ, ነበርሽ, ነበረ, ነበረች, ነበርን, ነበራችሁ, ነበሩ), plural markers (ዎች, ች, ኣች, ዎቹ, ቹ, ኣቹ), possessive suffixes (ዬ, ይ, ዋ, ው, ዎ, ያችሁ, ያቸው, ያት), and object suffixes (ኝ, ን, ህ, ሽ, ት, ው, ኣቸው, ኣት). After suffix removal, the stemmer looks for infixes – the strings ተ, እተ, እንተ, አስ, አሳ, አሰ – and removes the first occurrence if found inside the word. It then applies a final pass of plural marker removal. The entire process (prefix, suffix, infix, plural) is then repeated a second time to handle words with multiple layers of affixation, such as "አልተጻፈም", which becomes "ጻፍ". Through this exhaustive but deterministic set of rules, words like "ቤቶች" are reduced to "ቤት", and "ይጻፋል" to "ጻፍ", dramatically improving retrieval recall by conflating all morphological variants of a given root.

**Root / Affix Handling**

The final step performs two clean-up operations. First, it removes stopword – a predefined list of very common Amharic function words that carry little meaning, including ይህ (this), ያ (that), እና (and), ነው (is), ነበር (was), ሆነ (became), ሁሉ (all), ውስጥ (inside), እነዚህ (these), በ (in/at), ላይ (on), እንደ (like), ወደ (towards), ለ (for/to), ከ (from), and many others. These words are discarded entirely because they appear in almost every document and would only add noise to the index. Second, the system checks the length of the remaining stem: if the stem consists of more than one character, it is kept as the final index term; if it is only a single character (e.g., after aggressive stripping we might get 'አ'), the original word is kept as a safety net, though in practice most single-character words are already stop words. The output is a clean list of normalized stems, each representing a core concept, ready for indexing. For instance, the sentence "ይህ ቤት እና ቤቶች በአዲስ አበባ ውስጥ ይገኛሉ" becomes the stem list [ቤት, ቤት, አዲስ, አበባ, ገኛ] after stopword removal and stemming.

## 4.5 Document Indexing

Document indexing is the stage that follows preprocessing and plays a crucial role in enabling efficient and fast retrieval of scanned documents. At this stage, all processed information—such as normalized word tokens, extracted features, and cleaned word representations—is organized into a structured index. This index allows the retrieval system to quickly locate relevant documents without scanning the entire dataset, making it a core component of modern information retrieval systems (Zobel & Moffat, 2006; Manning et al., 2008).

an inverted index is used as the main indexing technique. An inverted index maps each term to the list of documents in which it appears, which significantly improves search speed and scalability. Before indexing, all preprocessing steps including noise removal, binarization, segmentation, normalization, and feature extraction are completed to ensure that only meaningful word representations are stored. For Amharic text, root-based normalization is applied to reduce morphological variations and group related word forms under a single entry. This helps reduce redundancy in the index and improves retrieval accuracy and efficiency. Recent research in information retrieval confirms that inverted indexing remains a fundamental and widely used structure, even in modern hybrid and large-scale retrieval systems (Dong et al., 2023).

For instance, consider the Amharic root word "ዘፈን" (meaning "sing"). A single document may contain numerous surface variants such as ዘፈነ, ዘፈንኩ, ዘፈናችሁ, ዘፈንሽ, ዘፈንክ, ዘፈኑ, ዘኻፈነ, አዘፈናቸው, ዘፍነዋል, ይዘፍናሉ, ትዘፍናለች, among others. Rather than indexing each variant individually, the system maps all forms to the root code "ዘፈን", significantly reducing index size and ensuring that queries using any variant can retrieve the relevant document.

The index construction process begins by matching segmented word features extracted from document images with representations stored in the feature corpus. After feature comparison, stop-word removal is applied to eliminate frequently occurring but semantically weak terms that do not contribute significantly to document meaning. Since not all words carry equal importance in representing content, a term-weighting approach is used, where more frequent and meaningful words are assigned higher weights to improve their contribution to document representation. This helps the system focus on informative terms while reducing the influence of less relevant ones, thereby improving retrieval efficiency and accuracy in the indexing process (Manning, Raghavan, & Schütze, 2008).

Common Amharic words such as connectors and function words (e.g., ይህ, ያ, እና) appear frequently in almost all documents but contribute little semantic meaning. Excluding these terms from the index prevents unnecessary inflation of database size and avoids compromising search results.

Each indexed word object is represented by a structured feature representation, encoding its visual and linguistic characteristics extracted during preprocessing. By storing normalized root word forms alongside these features, the system ensures that search queries can efficiently match relevant documents even when surface variations of words differ, improving retrieval consistency in morphologically rich languages (Manning, Raghavan, & Schütze, 2008).

## 4.6 Query Processor

The Query Processor serves as the primary interface between the user and the proposed Amharic Document Image Retrieval System (ADIRS). Its main role is to accept user queries and manage the retrieval of relevant document images in an efficient and user-friendly manner. In document image retrieval systems, maintaining consistency between query processing and document indexing is essential for achieving reliable retrieval performance (Manning, Raghavan, & Schütze, 2008).

To support this requirement, the query processor is implemented using a graphical user interface (GUI) that allows users to submit Amharic text queries in a natural and intuitive way. The interface is designed specifically for native Amharic users and does not require prior technical knowledge of OCR or retrieval mechanisms. GUI-based interaction models are widely adopted in information retrieval systems to improve usability and accessibility for end users (Baeza-Yates & Ribeiro-Neto, 2011).

The query processor also incorporates linguistic handling to address the morphological complexity of Amharic. Since Amharic is a morphologically rich language, a single root can generate multiple surface forms through inflection and derivation. To manage this variation, root-based normalization is applied so that different word forms are mapped to a shared underlying representation. This improves retrieval effectiveness by increasing the chance of matching relevant documents even when query and indexed terms differ, which is a common issue in low-resource language processing.

After feature extraction, the system performs similarity matching between the query representation and the indexed feature corpus. Documents are then ranked based on term importance and relevance scoring using frequency-based weighting and vector-space similarity principles. This ranking strategy ensures that the most relevant document images are retrieved first, improving both precision and user experience in the proposed system (Zhai, 2022; Liu et al., 2021).

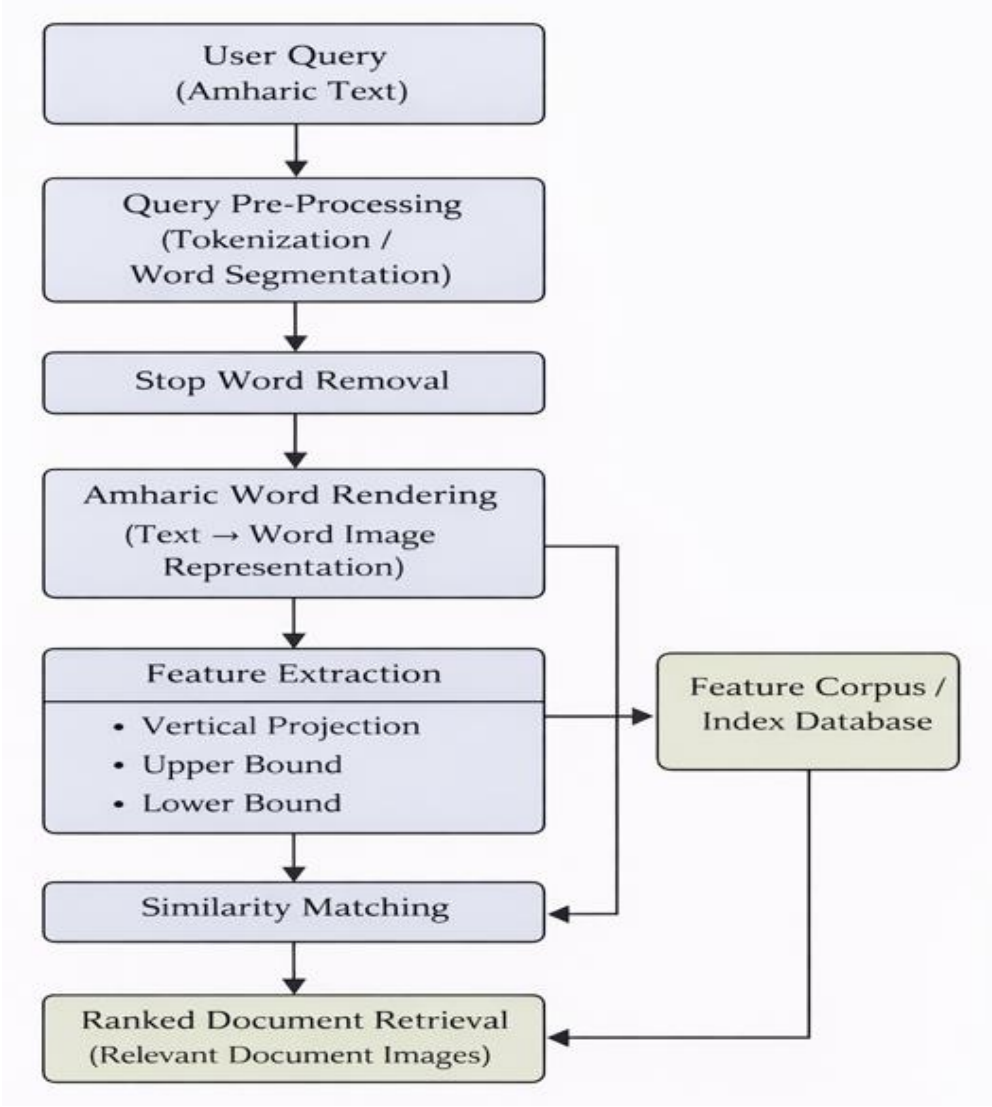



Figure 4.3 Query Processing flow diagram

The query processing component, illustrated in Figure 4.5, serves as the primary interface between the user and the proposed Amharic Document Image Retrieval System with the approach of NLP. Its main role is to interpret user input and transform it into a structured representation that the system can process efficiently. This module is designed specifically to handle the linguistic complexities of Amharic, a morphologically rich language with many high-frequency function words that contribute little to semantic meaning.

when a user enters an Amharic query, the system first performs word-level tokenization by splitting the input into individual terms using whitespace. Although Amharic naturally uses spaces between words, this step ensures consistent preprocessing before further analysis. The tokenized

query is then normalized and transformed into a structured representation so it matches the same feature space as the indexed documents, which is essential for accurate retrieval in information retrieval systems (Zhai, 2022).

Not all words in a query are equally informative for retrieval. High-frequency Amharic words such as ሆነ, ሁሉ, ነበር, ነው, ውስጥ, ይህ, ያ, and እነዚህ appear frequently in most documents but carry minimal semantic content. These words often function as articles, conjunctions, or prepositions. To reduce noise and improve retrieval precision, each segmented query word is compared against a predefined Amharic stop-word list developed for the system, and any matches are excluded from further processing.

After stop-word removal, the system applies morphological normalization, which is critical due to Amharic's high degree of inflection. Words may appear in multiple surface forms, yet users expect all variants of the same route to retrieve similar results. The proposed system performs stemming to reduce each word to its root form. For example, the variants ቤት, ቤቱ, ቤቶች, ቤታችን, ቤቶቻችን, ቤትሽ, and ቤትህ are all derived from the root ቤት. By mapping all surface forms to a single root, the system treats them equivalently during retrieval, enhancing both recall and precision (Baeza-Yates & Ribeiro-Neto, 2011).

Once the linguistic processing is complete, the remaining query words are converted into word images, enabling the application of the same OCR-driven preprocessing and feature extraction techniques used during indexing. Each word image undergoes segmentation, cropping, and feature extraction to produce feature vectors capturing vertical projection, upper and lower profiles, and normalized dimensions that are comparable to the feature corpus stored in the system. These feature vectors are then matched against the precomputed templates in the index database. The results identify documents containing relevant terms and generate a candidate set for ranking.

By integrating linguistic morphological processing with OCR-based image analysis and NLP techniques, the query processor bridges the gap between user input and the system's internal document representation. This hybrid approach enables the system to efficiently retrieve Amharic document images despite the challenges of morphological complexity and variation, ensuring accurate and robust search performance.

### 4.6.1 Searching Techniques

Searching within the proposed Amharic Document Image Retrieval plays a key role in both indexing and retrieval stages. During index construction, the system performs matching between segmented word image features extracted from Amharic documents and the stored feature representations in the lexicon. This search process is used to determine the most appropriate normalized root form of each word so that different surface variations are mapped to a single unified representation in the index. This is particularly important for Amharic, where rich morphology leads to multiple word forms derived from a single root, and failing to normalize them can reduce retrieval accuracy and increase redundancy in the index (Yeshambel et al., 2022).

The second purpose is user-oriented retrieval, where the system searches the index database to retrieve document images corresponding to the processed query terms. Each query word image is represented by a feature vector, which is compared against the vectors stored in the feature corpus. The cosine similarity measure is used to evaluate similarity, as described in Section 3.2.4. Cosine similarity is particularly suitable because it measures the angle between vectors, making it robust to differences in word length or feature scaling. This ensures accurate matching even when scanned word images vary slightly in size or shape (Baeza-Yates & Ribeiro-Neto, 2011).

When an exact match is found between a query term and a stored word image template, the system retrieves the document identifiers (IDs) associated with that template and temporarily stores them in a results table. This table preserves matched documents along with metadata, such as word occurrence frequency, which is later used during ranking. If no match is found, the system provides a user-friendly message indicating that no relevant documents were located.

It is important to note that during the search stage, additional preprocessing like cropping or resizing is unnecessary. These steps have already been applied to both indexed and query word images. Consequently, all feature vectors whether from the corpus or generated from the query share consistent dimensions and representation formats, ensuring efficient and reliable similarity computation.

```
import os
import json
import sys
from core.nlp import AmharicTextNormalizer
from core.index import InvertedIndex
from core.search import search

def load_config():
    with open("config.json", "r", encoding="utf-8") as f:
        return json.load(f)

def main():
    # Load configuration and index
    config = load_config()
    normalizer = AmharicTextNormalizer(config)
    index = InvertedIndex(config)
    if not index.load():
        print("❌ No index found. Please import and index documents first.")
        return

    print("\n" + "="*60)
    print("ADIRS SEARCHING TECHNIQUES DEMONSTRATION")
    print("="*60)
    print(f"Documents in index: {index.doc_count}")
    print(f"Unique stems: {len(index.index)}")
    print("\nExample stems in index:", list(index.index.keys())[:10])
    print("="*60)

    while True:
        query = input("\n🔍 Enter Amharic query (or 'quit' to exit): ").strip()
        if query.lower() == 'quit':
            break
        if not query:
            continue

        # Normalize query (stopword removal, stemming, homophone normalization)
        terms = normalizer.normalize_text(query)
        if not terms:
            print("   No meaningful terms after normalization.")
            continue

        print(f"   Normalized stems: {terms}")

        # Perform search (returns list of dicts with 'document' and 'score')
        results = search(terms, index, config)
```

```
        if not results:
            print("   No matching documents found.")
        else:
            print(f"\n    Found {len(results)} documents:")
            for i, res in enumerate(results, 1):
                print(f"      {i}. {res['document']} (score:
{res['score']:.4f})")

if __name__ == "__main__":
    main()
```

sample implementation source code for Searching

### 4..6.2 Matching experiment

Matching is a core component of the proposed Amharic Document Image Retrieval System, following closely after query formulation and document representation in the retrieval pipeline. Its primary objective is to determine how closely the features of a query word correspond to those of each word template stored in the feature corpus.

Unlike traditional OCR systems that rely on character-level segmentation, modern document image retrieval approaches treat the whole word as the basic matching unit. This is because character segmentation in document images is often error-prone, especially in complex scripts and noisy data. Instead, word images are compared using holistic visual features, making the system more robust to noise and variation. After detection, each word image is resized and aligned to a standard format before feature comparison to ensure consistency in matching, as commonly used in word spotting and document image retrieval systems (Gintis et al., 2017).

Similarity between word images is computed using cosine similarity, a standard technique in vector-based information retrieval. Each word image is represented as a feature vector capturing its morphological and structural properties, including vertical projections, upper and lower profiles, and normalized dimensions. When a query is submitted, the system constructs a feature vector for the query word and compares it against the template vectors stored in the index. The cosine similarity between vectors a and b is calculated as:

During index construction, if the similarity score between two-word vectors exceeds a predetermined threshold, the system groups them into the same word class, reducing redundancy and improving retrieval efficiency. In the retrieval stage, the Vector Space Model (VSM) is used

to compare the query vector with all document vectors in the database. Documents are then ranked according to their cosine similarity scores, ensuring that the most relevant documents appear at the top of the results.

By using word-level matching, cosine similarity, and the Vector Space Model (VSM), the proposed system is able to retrieve Amharic document images in an accurate and efficient way. In this approach, both queries and documents are represented as vectors, and similarity is measured based on how close they are in the vector space. This makes it easier for the system to identify relevant documents even when word forms are slightly different or when the input data contains noise. Overall, this method improves the system's ability to handle variation in Amharic text and supports more reliable retrieval results (Manning, Raghavan, & Schütze, 2008; Zhai, 2022).

```
import os
import json
import numpy as np
from core.nlp import AmharicTextNormalizer
from core.index import InvertedIndex
from core.search import search  # uses cosine similarity internally

def load_config():
    with open("config.json", "r", encoding="utf-8") as f:
        return json.load(f)

def main():
    config = load_config()
    normalizer = AmharicTextNormalizer(config)
    index = InvertedIndex(config)
    if not index.load():
        print("❌ No index found. Please import and index documents first.")
        return

    print("\n" + "="*60)
    print("ADIRS MATCHING TECHNIQUE DEMONSTRATION")
    print("(TF-IDF + Cosine Similarity)")
    print("="*60)
    print(f"Total documents in index: {index.doc_count}")
    print(f"Unique stems: {len(index.index)}")
    print("="*60)

    while True:
        query = input("\n🔍 Enter Amharic query (or 'quit'): ").strip()
        if query.lower() == 'quit':
```

```
            break
        if not query:
            continue

        # 1. Normalize query
        terms = normalizer.normalize_text(query)
        if not terms:
            print("   No meaningful terms after normalization.")
            continue
        print(f"   Query stems: {terms}")

        # 2. Build query vector (TF-IDF)
        q_vec = {}
        for t in set(terms):
            df = index.get_term_df(t)
            idf = np.log((index.doc_count + 1) / (df + 1)) + 1
            q_vec[t] = idf  # term frequency in query = 1
        print(f"   Query vector (IDF weights): {q_vec}")

        # 3. Get candidate documents (those sharing at least one stem)
        candidates = index.get_candidate_docs(terms)
        if not candidates:
            print("   No candidate documents found.")
            continue

        print(f"   Candidate documents: {candidates}")

        # 4. For each candidate, compute matching score (cosine similarity)
        scores = []
        for doc_id in candidates:
            d_vec = index.get_tfidf_vector(doc_id, terms)  # TF-IDF for document
            all_terms = set(q_vec.keys()) | set(d_vec.keys())
            if not all_terms:
                continue
            # Convert to aligned arrays
            q_arr = np.array([q_vec.get(t, 0) for t in all_terms])
            d_arr = np.array([d_vec.get(t, 0) for t in all_terms])
            # Cosine similarity
            norm_q = np.linalg.norm(q_arr)
            norm_d = np.linalg.norm(d_arr)
            sim = np.dot(q_arr, d_arr) / (norm_q * norm_d) if norm_q and norm_d
else 0
            scores.append((doc_id, sim))

        # Sort by similarity descending
```

```
        scores.sort(key=lambda x: x[1], reverse=True)

        print("\n   Matching results (ranked by cosine similarity):")
        for doc_id, sim in scores[:10]:
            print(f"      {doc_id}: similarity = {sim:.4f}")

if __name__ == "__main__":
    main()
```

Figure sample implementation source code for matching

### 4.7 Ranking search result

Once the searchable data structure has been constructed and the system has successfully transformed raw document content into an organized index, the next essential step is determining how the system responds to a user's query. This process involves identifying which documents from the collection are relevant and presenting them in an order that reflects their likelihood of satisfying the user's information need. The act of matching a query to indexed items is known as searching, and it is one of the central operations of any information retrieval (IR) system. In the context of rule-based Amharic image retrieval, ranking becomes even more significant because Amharic documents often vary in structure, orthography, and morphological patterns. As a result, once the rule-based linguistic processing identifies candidate terms or features from the scanned image, the system must calculate how closely each document aligns with the user's query terms.

Modern information retrieval (IR) systems use weighted term representations to measure the similarity between user queries and stored documents. These weights help the system rank documents according to their relevance, ensuring that the most useful results appear first. In large Amharic document collections, ranking is especially important for improving retrieval efficiency and user access to relevant information. Many IR systems apply TF–IDF and vector-space similarity methods to evaluate both term frequency and term importance during retrieval (Manning, Raghavan, & Schütze, 2008; Yeshambel, Mothe, & Assabie, 2022).

By computing similarity scores using weighted terms, the system can effectively determine which documents provide the closest semantic match to the user's needs and present them in descending ng order of relevance.

## 4.8 Contribution of Research

The main contribution of this research is the design and implementation of a rule-based retrieval system that integrates OCR, NLP, and information retrieval techniques into a single framework. The system improves the accessibility of scanned Amharic documents by converting document images into searchable text through OCR. The extracted text is then processed using NLP techniques such as tokenization, normalization, and stop-word removal to improve retrieval performance. An inverted indexing mechanism is used to organize the processed text and enable efficient document searching. In addition, a user-friendly graphical interface was developed to support document uploading, indexing, and retrieval. The system was evaluated using Precision, Recall, F-score, and Accuracy metrics, demonstrating its effectiveness in retrieving relevant documents from scanned Amharic document collections.

### Novelty of the Study

The novelty of this study lies in the integration of OCR, NLP, and rule-based retrieval techniques into a unified system specifically designed for Amharic scanned documents. Unlike many previous studies that addressed these components separately, this research combines them into a complete framework that supports the entire process from scanned document input to document retrieval. This provides a practical solution for institutions that manage large collections of scanned Amharic documents and need an efficient way to search and access information.

## 4.10 Experimental Resources

The experimental evaluation of the proposed System was conducted using both laptop and desktop computers with moderate hardware configurations. The laptop should Pentium® Dual-Core processor operating at 2.5 GHz with 4 GB RAM. This is used during implementation, testing, and evaluation of the system.

Although the hardware resources used during implementation were relatively available, they were sufficient for implementing and evaluating the proposed Amharic scanned document retrieval framework. This demonstrates that the developed system can operate effectively even in low-resource environments, making it suitable for practical deployment in libraries, educational institutions, government offices, and organizations that manage large collections of scanned Amharic documents.

The implementation environment was developed under the Windows operating system using Python as the primary programming language. Python was selected because of its simplicity, flexibility, and extensive support for image processing, OCR, and natural language processing applications. Several software libraries and tools were integrated into the system implementation. OpenCV was used for image preprocessing, noise removal, binarization, segmentation, and feature extraction operations. Tesseract OCR with Amharic language support was employed to extract textual information from scanned Amharic document images. In addition, NumPy and Scikit-learn libraries were utilized for vector representation, TF-IDF ranking, cosine similarity matching, and numerical computations.

The developed prototype integrates multiple intelligent modules including image preprocessing, line and word segmentation, OCR processing, Amharic NLP analysis, indexing, hybrid matching, ranking, and retrieval modules. The NLP component applies based on rule linguistic processing techniques such as tokenization, stop-word removal, normalization, and stemming to handle the rich morphology and orthographic variations of the Amharic language. Extensive experiments were conducted using different Amharic query terms and document collections to evaluate OCR accuracy, retrieval effectiveness, matching precision, ranking quality, and overall system performance.

### 4.10. Data Sets used

The experiments conducted using four categories of scanned Amharic document image collections: printed documents, magazines, newspapers, and books. These datasets were selected because they represent common real-world Amharic documents written using different font types, font sizes, writing styles, and formatting structures. The use of diverse document categories enabled the proposed retrieval system to be evaluated under realistic and challenging conditions.

All document images were scanned and stored in grayscale format (see Appendix II). Grayscale images were preferred because they simplify preprocessing operations and improve computational efficiency during noise removal, segmentation, OCR processing, and feature extraction stages. The collected datasets include both clean and moderately noisy scanned document images to evaluate the robustness of the proposed system under different image quality conditions.

The proposed system is tested using multiple Amharic query terms and different retrieval scenarios. During experimentation, the system evaluated segmentation accuracy, OCR

performance, NLP effectiveness, hybrid matching efficiency, ranking quality, and overall retrieval performance. The experimental dataset provided a comprehensive environment for validating the effectiveness and robustness of the proposed Amharic scanned document retrieval system with the approach of NLP and OCR.

### 4.11 Test Result

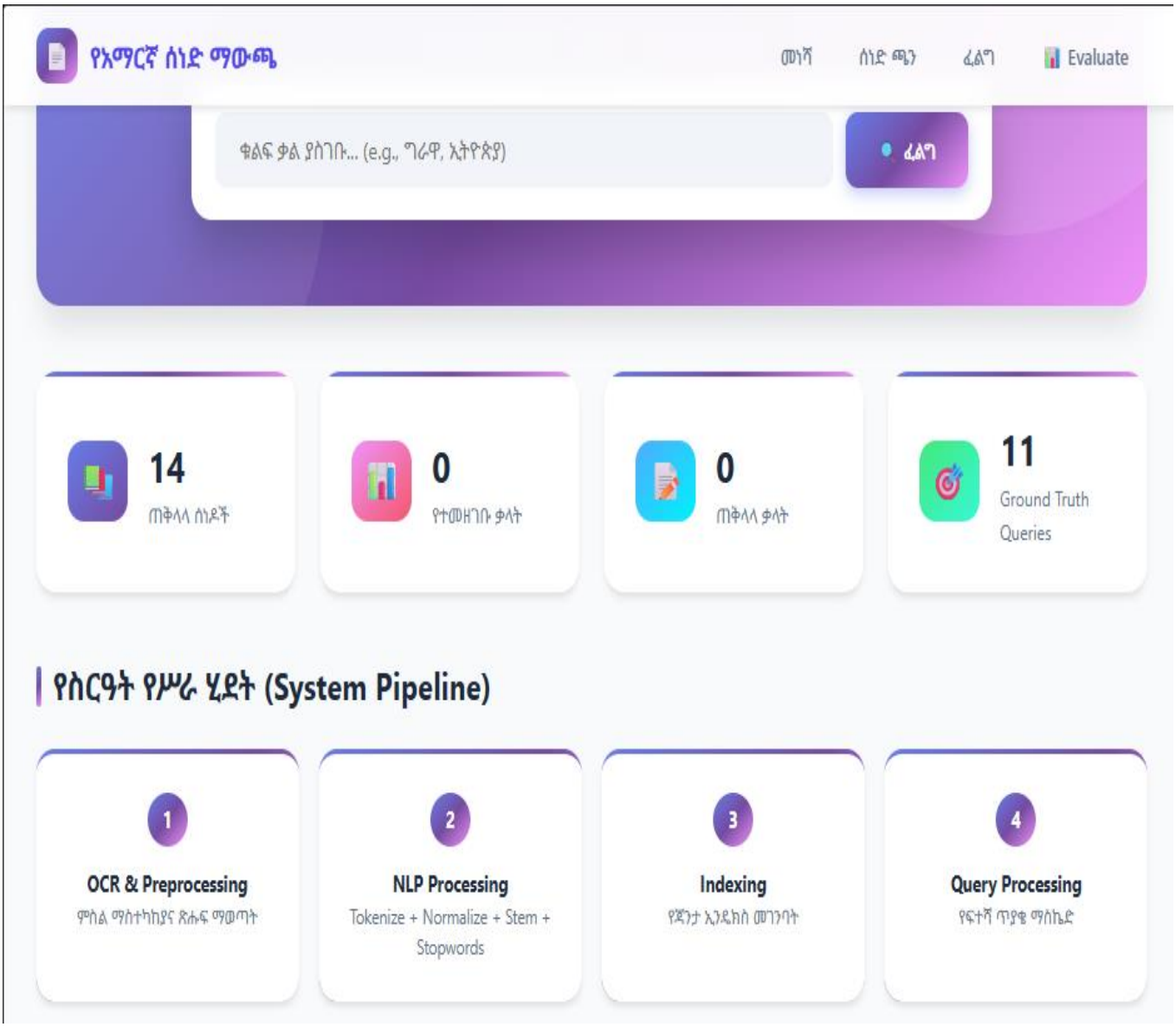



Figure 4.4 home page user interface

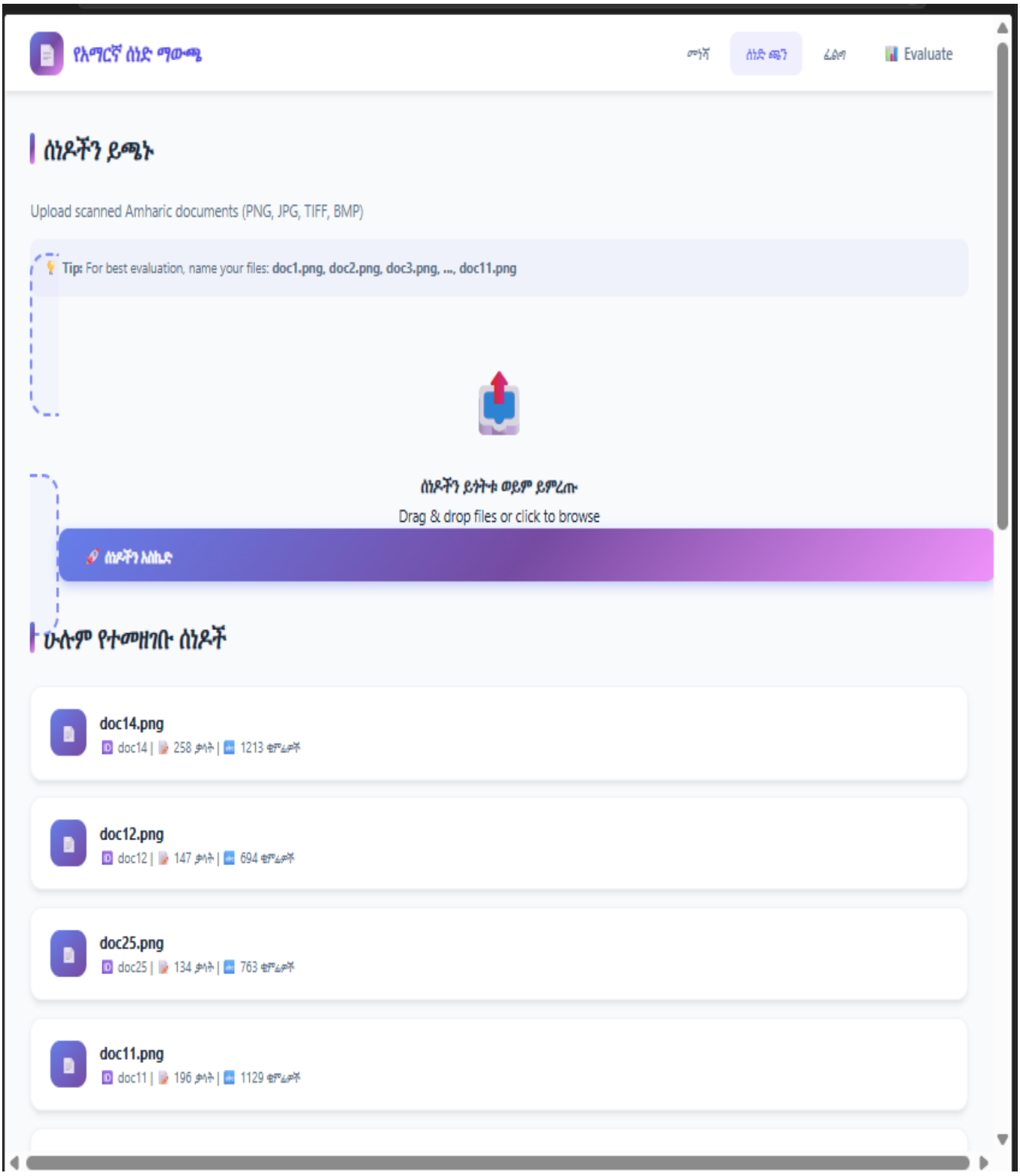



Figure 4.5 user interface shows uploading scanned document

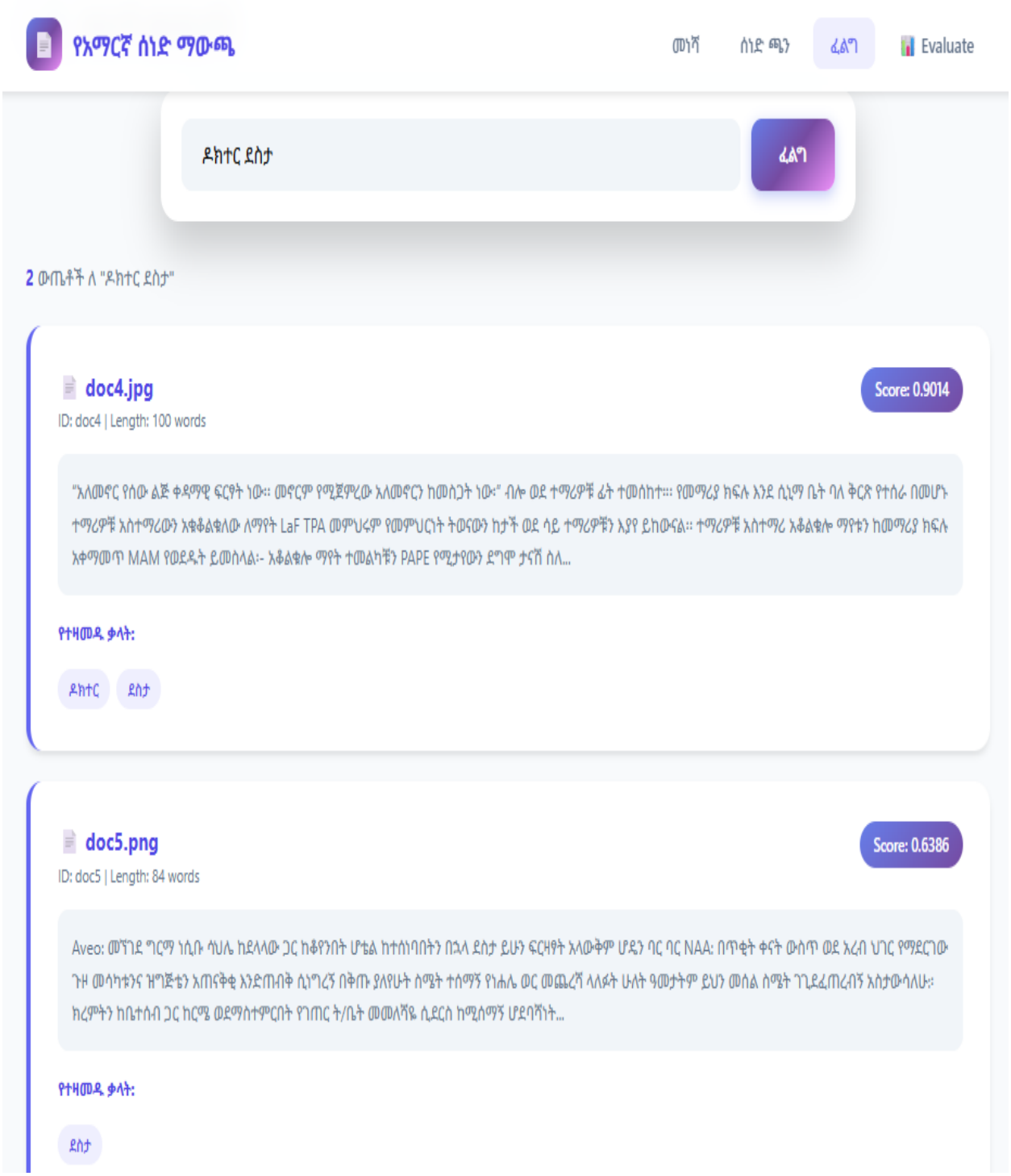



Figure 4.6 user interface shows searching and result

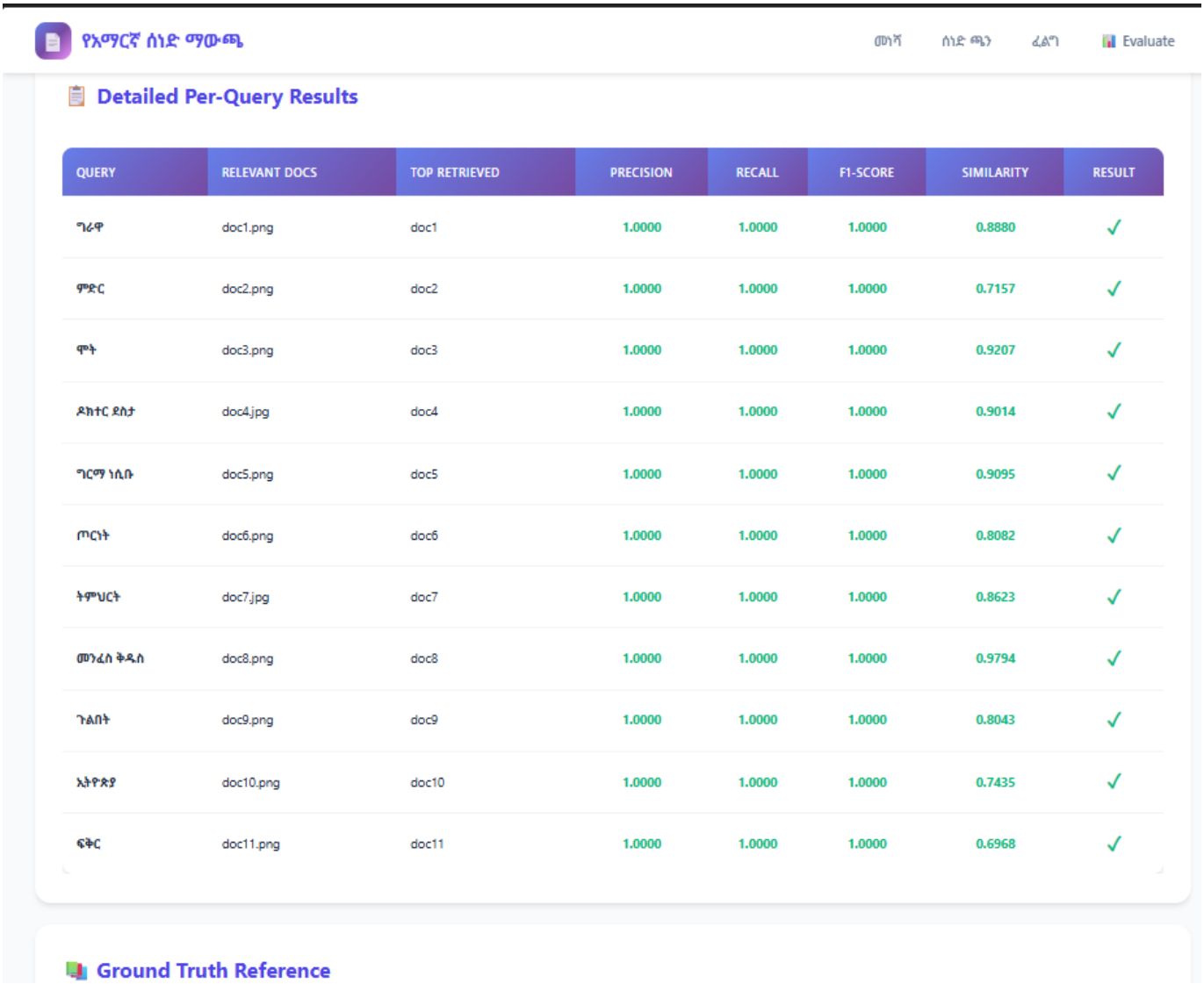



Figure 4.7 user interface shows overall retrieval performance and result

**Summary for the evaluation table**

| Metrics | Average |
|---|---|
| Precision | 1.00 |
| Recall | 1.00 |
| F1 | 1.00 |
| Accuracy | 1.00 |
| Similarity | 0.8391 |
| Time | 1.21ms |

4.1 Result Summary table

This study developed an Amharic scanned document retrieval system that integrates image preprocessing, Optical Character Recognition (OCR), and rule-based Natural Language Processing (NLP) techniques to improve the retrieval of scanned Amharic documents. The system was specifically designed to address challenges such as poor scan quality, document noise, OCR extraction errors, and the complex morphological structure of the Amharic language. To improve text extraction quality, the framework incorporates preprocessing techniques such as noise removal, image enhancement, adaptive binarization, and segmentation before OCR processing. In addition, the system applies Amharic-aware normalization, stemming, stop-word removal, and indexing mechanisms to enhance query matching and retrieval effectiveness.

The retrieval framework further integrates inverted indexing, TF-IDF weighting, and cosine similarity ranking to identify and rank relevant documents efficiently. Experimental evaluations demonstrated that the proposed system performs effectively in extracting text from different types of scanned Amharic documents and provides reliable retrieval performance with improved accuracy and query response efficiency compared to conventional OCR-based retrieval approaches. In general, the study contributes a practical, scalable, and linguistically informed framework that can support efficient access, management, and retrieval of large collections of Amharic digital documents and archives.

# CHAPTER FIVE

# CONCLUSIONS AND RECOMMENDATIONS

## 5.1 Conclusions

Scanned images document has become an indispensable source of information in libraries, government offices, courts, and historical archives. In Ethiopia, a vast number of valuable documents administrative records, court decisions, educational materials, and cultural heritage texts exist only as scanned images. Accessing the information contained in these images efficiently is a major challenge, especially for the Amharic language, which has a complex syllabic script and a very rich morphological system.

In this thesis, an Amharic scanned document retrieval system based on Natural Language Processing (NLP) is proposed to address the limitations of existing retrieval approaches for degraded and morphologically less complex Amharic documents. The system integrates intelligent image preprocessing, adaptive Optical Character Recognition (OCR), and linguistic processing framework specifically designed for the Amharic language. Unlike conventional OCR retrieval systems, the proposed approach combines orthographic normalization, affix-aware morphological analysis, stop-word filtering, and Amharic word synthesis to improve retrieval effectiveness across different word forms and noisy scanned texts.

The system follows a clear pipeline: first, scanned images are cleaned using a median filter for noise removal and Otsu's method for binarization. Second, Tesseract OCR with the Amharic language model converts the cleaned images into editable text. Third, Amharic NLP module performs three critical transformations: orthographic normalization (mapping homophones such as ሐ, ሓ, ኀ → ሀ), morphological stemming (removing a comprehensive list of prefixes, suffixes, infixes, and circumfixes), and stopword removal (filtering out common function words like ይህ, እና, ነው, በ, ለ, ከ). This process reduces all morphological variants of a word to a common stem, dramatically improving recall. Fourth, an inverted index with TF-IDF weighting is built, and retrieval is performed using cosine similarity. Finally, a graphical user interface allows users to import documents, search, evaluate, and test each component.

These results confirm that NLP approach, combined with OCR and classical information retrieval techniques, can significantly improve retrieval accuracy for scanned Amharic documents.

Although the proposed system produced encouraging and reliable results, some challenges still remain. OCR errors are occasionally observed when the system processes low-quality scanned documents, particularly those affected by skewed pages, faded printing, bleed-through noise, shadows, or very small font sizes These issues sometimes reduce the quality of the extracted text and can influence retrieval performance. Additionally, the current preprocessing stage does not yet include more advanced correction methods such as automatic deskewing, illumination balancing, or intelligent page restoration, which could further improve OCR accuracy for difficult document conditions.

The morphological framework was carefully designed to handle many common Amharic linguistic patterns; but, due to the richness and complexity of the language, some rare or highly irregular affix structures may still not be fully covered. Also, the experimental evaluation was conducted on a relatively limited scanned document.

## 5.2 Recommendations

Based on the findings and limitations of this study, future work is needed because the proposed system was developed and tested on a relatively small collection of clean printed documents, which limits its performance on real-world noisy, degraded, and large-scale datasets. The current preprocessing methods (median filtering and Otsu binarization) are effective for clear scans but struggle with skewed, unevenly illuminated, or heavily degraded pages, so future improvements should include adaptive thresholding (such as Sauvola), deskewing techniques, morphological operations to repair broken characters, and possibly deep learning-based denoising for more challenging documents. In addition, the rule-based Amharic morphological analyzer performs well for common patterns but does not fully capture rare or irregular word formations due to limited linguistic coverage, so expanding the rule set using larger corpora and combining rule-based and statistical or finite-state approaches would improve robustness. Moreover, while TF-IDF with cosine similarity provides a strong baseline for retrieval, more advanced ranking models like BM25 or learning-to-rank methods could further improve accuracy, especially in larger datasets. Finally, since the system has only been evaluated on a small dataset, future work should focus on scalability through incremental indexing, index compression, and parallel processing to ensure efficient performance as the document collection grows.

# REFERENCES


Hersh, W. R. (2023). Search still matters: Information retrieval in the era of generative

Manning, C. D., Raghavan, P., & Schütze, H. (2018). Introduction to information retrieval. Cambridge University Press.

Wende, F. (2015). Optical character recognition and document image analysis: A survey of techniques and challenges. International Journal of Computer Applications, 120(6), 1–6.

Smith, R. (2016). An overview of the Tesseract OCR engine. In Proceedings of the International Conference on Document Analysis and Recognition (ICDAR) (pp. 629–633). IEEE.

Jurafsky, D., & Martin, J. H. (2023). Speech and language processing (3rd ed. draft). Stanford University. https://web.stanford.edu/~jurafsky/slp3/

Belay, B., Habtegebrial, T., Meshesha, M., Liwicki, M., Belay, G., & Stricker, D. (2020). Amharic OCR: An end-to-end learning approach for text-line image recognition. Applied Sciences, 10(3), 1117.

Yeshambel, T., Mothe, J., & Assabie, Y. (2022). Amharic adhoc information retrieval system based on morphological features. Applied Sciences, 12(3), 1294.

Yeshambel, T., Mothe, J., & Assabie, Y. (2023). Learned text representation for Amharic information retrieval and NLP. Information, 14(3), 195.

Tonja, A. L., et al. (2024). EthioLLM: Multilingual large language models for Ethiopian languages. arXiv.

Gezmu, A. M., Seyoum, B. E., Gasser, M., & Nürnberger, A. (2021). Contemporary Amharic corpus: Automatically morpho-syntactically tagged Amharic corpus. arXiv.

Abate, S. T., Melese, M., Meshesha, M., Assabie, Y., & Schwenk, H. (2018). Large-scale English–Amharic parallel corpus for statistical machine translation. In Proceedings of LREC 2018.

Tsegay, S., Yimam, S. M., & Biemann, C. (2020). Exploring NLP resources and preprocessing techniques for Ethiopian languages. Proceedings of recent African language NLP studies.

Abate, S. T., Melese, M., Meshesha, M., Assabie, Y., & Schwenk, H. (2018). Large-scale English–Amharic parallel corpus for statistical machine translation. Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018).

Amsalu, A., & Demeke, G. (2006). Learning morphological rules for Amharic verbs. Proceedings of the Ethiopian Language Technology research studies.

Yeshambel, T., Mothe, J., & Assabie, Y. (2022). Amharic ad hoc information retrieval system based on morphological features. Applied Sciences, 12(3), 1294.

Unicode Consortium. (2024). The Unicode Standard, Chapter 19: Ethiopic script. https://www.unicode.org/versions/latest/core-spec/chapter-19/

Addis Ababa University. (2010). Amharic language and computational linguistics materials. Addis Ababa University, Department of Linguistics.

Yimam, S. M., Gashaw, I., Alemu, H., & Biemann, C. (2017). Exploring NLP for Ethiopian languages: preprocessing and semantic challenges. Proceedings of African Language Technology Studies.

Mitkov, R. (2002). Anaphora resolution. Pearson Education.

Croft, W. B., Metzler, D., & Strohman, T. (2009). Search engines: Information retrieval in practice. Addison-Wesley.

Hutchins, W. J. (1986). Machine translation: Past, present, future. Ellis Horwood.

Manning, C. D., Raghavan, P., & Schütze, H. (2008). Introduction to information retrieval. Cambridge University Press.

Porter, M. F. (1980). An algorithm for suffix stripping. Program, 14(3), 130–137.

Devlin, J., Chang, M. W., Lee, K., & Toutanova, K. (2019). BERT: Pre-training of deep bidirectional transformers for language understanding. NAACL.

Navarro, G. (2022). Compact data structures: A practical approach. Cambridge University Press.

Büttcher, S., Clarke, C. L. A., & Cormack, G. V. (2016). Information retrieval: Implementing and evaluating search engines (2nd ed.). MIT Press.

Zhai, C., & Massung, S. (2016). Text data management and analysis: A practical introduction to information retrieval and text mining. ACM Books.

Baeza-Yates, R., & Ribeiro-Neto, B. (2011). Modern information retrieval: The concepts and technology behind search (2nd ed.). Addison-Wesley

Robertson, S., & Zaragoza, H. (2009). The probabilistic relevance framework: BM25 and beyond. Foundations and Trends in Information Retrieval, 3(4), 333–389.

Hirpassa. (2024). [Title of your source if available].

Robertson, S., & Zaragoza, H. (2009). The probabilistic relevance framework: BM25 and beyond. Foundations and Trends in Information Retrieval, 3(4), 333–389.

Croft, W. B., Metzler, D., & Strohman, T. (2015). Search engines: Information retrieval in practice. Addison-Wesley.

Manning, C. D., Raghavan, P., & Schütze, H. (2016). Introduction to information retrieval. Cambridge University Press.

Sauvola, J., & Pietikäinen, M. (2000). Adaptive document image binarization. Pattern Recognition, 33(2), 225–236

Bukhari, S. S., Shafait, F., & Breuel, T. M. (2016). High-performance adaptive binarization of historical document images. Pattern Recognition, 60, 590–605.

Gonzalez, R. C., & Woods, R. E. (2018). Digital image processing (4th ed.). Pearson

Breuel, T. M. (2008). The OCRopus open source OCR system. Proceedings of SPIE Document Recognition and Retrieval XV.

Rigaud, C., Kacem, A., & Ogier, J. M. (2019). A survey of document image analysis techniques for historical documents. Pattern Recognition Letters, 121, 1–11

Likforman-Sulem, L., Zahour, A., & Taconet, B. (2007). Text line segmentation of historical documents: A survey. International Journal on Document Analysis and Recognition, 9(2–4), 123–138.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., … Duchesnay, E. (2011). Scikit-learn: Machine learning in Python. Journal of Machine Learning Research, 12, 2825–2830.

Plötz, T., & Roth, S. (2011). What makes text recognition different? Proceedings of the International Conference on Document Analysis and Recognition (ICDAR), 1–6.

Jain, A. K., Duin, R. P. W., & Mao, J. (2000). Statistical pattern recognition: A review. IEEE Transactions on Pattern Analysis and Machine Intelligence, 22(1), 4–37.

Singhal, A. (2001). Modern information retrieval: A brief overview. IEEE Data Engineering Bulletin, 24(4), 35–43.

Gallet, B., & Gowanlock, M. (2022). Leveraging GPU tensor cores for double precision Euclidean distance calculations. In Proceedings of the IEEE 29th International Conference on High Performance Computing, Data, and Analytics (HiPC 2022), 135–144.

Morelli, L., Karami, A., Menna, F., & Remondino, F. (2022). Orientation of images with low contrast textures and transparent objects. International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences, XLVIII-2/W2-2022, 77–84.

Alzoubi, Y. I., Topcu, A. E., & Ozdemir, E. (2024). Enhancing document image retrieval in education: Leveraging ensemble-based document image retrieval systems for improved precision. Applied Sciences, 14(2), 751.

Balasubramanian, A., Meshesha, M., & Jawahar, C. V. (2006). Retrieval from document image collections. In International Workshop on Document Analysis Systems.

agoris, K., Amanatiadis, A., & Pratikakis, I. (2021). Word spotting as a service: An unsupervised and segmentation-free framework for handwritten documents. Journal of Imaging, 7(12), 278.

Worku, M. (2009). Amharic document image retrieval without explicit recognition (Master's thesis). Addis Ababa University. 45ff3230f889

Anand, R., et al. (2007). Content-based word image retrieval using locality-sensitive hashing. International Journal of Document Analysis and Recognition.

u, S., Tan, C. L., & others (2008). Document image retrieval through word shape coding. Pattern Recognition, 41(5), 1799–1809.

Marinai, G., Marino, E., & Soda, G. (2005–2007 works on document image analysis and retrieval systems in digital libraries)

Zagoris, K., Pratikakis, I., & Gatos, B. (2010)."A word spotting framework for historical document images based on region-based features"Pattern Recognition, 43(11), 3883–3894.

Abreham Gebresadik (2010). Searching in Amharic Document Image Corpus: Master's thesis, Department of Information Science, Addis Ababa University, Ethiopia.

Gasser, M. (2011). HornMorpho: A system for morphological processing of Amharic, Oromo, and Tigrinya.

Mekonnen, A., Seyoum, B. E., Gasser, M., & Nürnberger, A. (2021). Contemporary Amharic corpus: Automatically morpho-syntactically tagged Amharic corpus. arXiv:2106.07241.

smith, R. (2007). An overview of the Tesseract OCR engine. Proceedings of the Ninth International Conference on Document Analysis and Recognition (ICDAR 2007), 629–633.

Abdurahman, F., Sisay, E., & Fante, K. A. (2021). AHWR-Net: Offline handwritten Amharic word recognition using convolutional recurrent neural network. SN Applied Sciences, 3, 760.

Feyisa, T., & Hailu, S. (2023). Corpus generation to develop Amharic morphological segmenter. International Journal of Advanced Computer Science and Applications, 14(9).

Zobel, J., & Moffat, A. (2006). Inverted files for text search engines. ACM Computing Surveys, 38(2).

Dong, S., Goldstein, J., & Yang, G. H. (2023). SEINE: Segment-based indexing for neural information retrieval. arXiv preprint arXiv:2311.15923.

Yang, Z., Zuo, S., Zhou, Y., He, J., & Shi, J. (2024).A review of document image binarization: Main techniques, challenges and trends.Electronics, 13(7), 1394.

Zhai, C. (2022). Statistical language models for information retrieval. Foundations and Trends in Information Retrieval.

Liu, X., Chen, Q., & Wang, H. (2021). Neural ranking models for information retrieval: A survey. Information Processing & Management, 58(5), 102563.

Giotis, A. P., Sfikas, G., Gatos, B., & Nikou, C. (2017). A survey of document image word spotting techniques. Pattern Recognition, 68, 310–332.

## APPENDIX I: The Amharic Character and their Roman counter parts

| | Ge'ez<br>ä | ka'eb<br>u | Salis<br>ī | Rab'e<br>a | Hamis<br>é | Sadis<br>i | Sab'e<br>o |
|---|---|---|---|---|---|---|---|
| h | ሀ | ሁ | ሂ | ሃ | ሄ | ህ | ሆ |
| l | ለ | ሉ | ሊ | ላ | ሌ | ል | ሎ |
| h | ሐ | ሑ | ሒ | ሓ | ሔ | ሕ | ሖ |
| m | መ | ሙ | ሚ | ማ | ሜ | ም | ሞ |
| s | ሠ | ሡ | ሢ | ሣ | ሤ | ሥ | ሦ |
| r | ረ | ሩ | ሪ | ራ | ሬ | ር | ሮ |
| s | ሰ | ሱ | ሲ | ሳ | ሴ | ስ | ሶ |
| s | ሸ | ሹ | ሺ | ሻ | ሼ | ሽ | ሾ |
| q | ቀ | ቁ | ቂ | ቃ | ቄ | ቅ | ቆ |
| b | በ | ቡ | ቢ | ባ | ቤ | ብ | ቦ |
| v | ቨ | ቩ | ቪ | ቫ | ቬ | ቭ | ቮ |
| t | ተ | ቱ | ቲ | ታ | ቴ | ት | ቶ |
| c | ቸ | ቹ | ቺ | ቻ | ቼ | ች | ቾ |

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| h | ሃ | ሑ | ሒ | ሓ | ሔ | ሕ | ሖ |
| n | ለ | ሉ | ሊ | ላ | ሌ | ል | ሎ |
| n | ኘ | ኙ | ኚ | ኛ | ኜ | ኝ | ኞ |
| x | አ | ኡ | ኢ | ኣ | ኤ | እ | ኦ |
| k | ከ | ኩ | ኪ | ካ | ኬ | ክ | ኮ |
| x | ኸ | ኹ | ኺ | ኻ | ኼ | ኽ | ኾ |
| w | ወ | ዉ | ዊ | ዋ | ዌ | ው | ዎ |
| z | ዘ | ዙ | ዚ | ዛ | ዜ | ዝ | ዞ |
| z | ዠ | ዡ | ዢ | ዣ | ዤ | ዥ | ዦ |
| y | የ | ዩ | ዪ | ያ | ዬ | ይ | ዮ |
| d | ደ | ዱ | ዲ | ዳ | ዴ | ድ | ዶ |
| j | ጀ | ጁ | ጂ | ጃ | ጄ | ጅ | ጆ |
| g | ገ | ጉ | ጊ | ጋ | ጌ | ግ | ጎ |
| t | ጠ | ጡ | ጢ | ጣ | ጤ | ጥ | ጦ |
| c | ጨ | ጩ | ጪ | ጫ | ጬ | ጭ | ጮ |
| p | ጰ | ጱ | ጲ | ጳ | ጴ | ጵ | ጶ |

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| s | ጸ | ጹ | ጺ | ጻ | ጼ | ጽ | ጾ |
| s | ፀ | ፁ | ፂ | ፃ | ፄ | ፅ | ፆ |
| f | ፈ | ፉ | ፊ | ፋ | ፌ | ፍ | ፎ |
| p | ፐ | ፑ | ፒ | ፓ | ፔ | ፕ | ፖ |

## Appendix II: sample data from different sources



# ባለ አራት የሰው ሰራሽ ጉልበት

የሰውነት አካልን ማጣት፣ በተለይም እግርን ከጉልበት በላይ ማጣት (Transfemoral amputation) በአንድ ሰው የዕለት ተዕለት ሕይወት፣ የእንቅስቃሴ ነፃነት እና ሥነ-ልቦና ላይ ትልቅ ተጽዕኖ ያሳድራል። ይሁን እንጂ፣ በዘመናዊ የሕክምና ቴክኖሎጂ እና የሰው ሰራሽ አካላት (Prosthetics) እድገት ምክንያት ይህንን ከፍተት በከፍተኛ ደረጃ መመላት ተችሏል።

ከእነዚህ የቴክኖሎጂ ፈጠራዎች መካከል ከፍተኛ እውቅናን ያተረፈው "ባለ አራት-አሞሌ የሰው ሠራሽ ጉልበት መገጣጠሚያ" (4-Bar Prosthetic Knee Joint) በመባል የሚታወቀው ነው። ይህ ቴክኖሎጂ የተፈጥሮን ጉልበት እንቅስቃሴ በተሻለ ሁኔታ ለመኮረጅ የተቀየሰ ሲሆን፣ ለተጠቃሚዎች ከፍተኛ መረጋጋትን እና ምቾትን ይሰጣል። ይህ አርቲክል የዚህን ልዩ የጉልበት መገጣጠሚያ አሠራር፣ ጥቅሞች፣ ጉዳቶች፣ እና ለየትኞቹ ተጠቃሚዎች ይበልጥ ተስማሚ እንደሆነ በዝርዝር ይዳስሳል።

### የ4-ባር የጉልበት መገጣጠሚያ አሠራር (Mechanism)

በአጠቃላይ የሰው ሠራሽ ጉልበቶች በሁለት ዋና ዋና ከፍሎች ይከፈላሉ። ባለ አንድ-ዘንግ (Single-axis) እና ብዙ-ዘንግ ያላቸው (Polycentric) በመባል ይታወቃሉ። ባለ 4-አሞሌ (4-Bar) ጉልበት ከብዙ-ዘንግ ጉልበቶች ምድብ ውስጥ የሚመደብ ሲሆን፣ አሠራሩም አራት የተለያዩ መገጣጠሚያዎችን በማገናኘት የሚፈጠር ውስብስብ ግን እጅግ ጠቃሚ የሆነ የሜካኒክስ ሕግን ይከተላል።

የሰው ልጅ የተፈጥሮ ጉልበት በሚታጠፍበት እና በሚዘረጋበት ጊዜ በአንድ ቋሚ ነጥብ (ዘንግ) ላይ ብቻ አይሽከረከርም። ቡተቃራኒው፣ የማሽከርከሪያ ማዕከሉ (Center of rotation) እንደ ጉልበቱ አቀማመጥ ይለዋወጣል። የ4-ባር መገጣጠሚያ ይህንን የተፈጥሮ ሂደት በትክክል ለማስቀመጥ ያስችላል።

መገጣጠሚያው በሚንቀሳቀስበት ጊዜ የሽከርከሪት ማዕከሉ (Instantaneous Center of Rotation - ICR) ከጉልበቱ በላይ እና ወደ ኋላ ይሸጋል። ይህ ማለት ተጠቃሚው ክብደቱን በእግሩ ላይ ሲያሳርፍ (Stance phase) ጉልበቱ ወደ ፊት እንዳይታጠፍ እና ጠንካራ መረጋጋት እንዲኖረው ያደርጋል። በሌላ በኩል ደግሞ እግሩን ለማንሳት እና ለመራመድ ሲዘጋጅ መገጣጠሚያው የክብደት ማዕከሉን በመቀየር በቀላሉ ይታጠፋል።

### የባለ አራት የጉልበት መገጣጠሚያ ዋና ዋና ጥቅሞች

የዚህ አይነቱ የሰው ሠራሽ ጉልበት ለተጠቃሚዎች ከሚሰጣቸው በርካታ ጥቅሞች መካከል ዋና ዋናዎቹ የሚከተሉት ናቸው። ከፍተኛ መረጋጋት (High Stability) ከሌሎች የሰው ሠራሽ ጉልበቶች ጋር ሲነጻጸር፣ የ4-ባር ጉልበት ተጠቃሚው ቆሞ በሚቆዝበት ጊዜ ወይም ክብደቱን በሚያሳርፍበት ጊዜ እጅግ የላቀ መረጋጋትን ይሰጣል። የሽከርከሪት ማዕከሉ አቀማመጥ ተጠቃሚው ሳያስበው ጉልበቱ እንዳይታጠፍ እና እንዳይወድቅ ይከላከላል። ይህ ቡተለይ ገና አዲስ ለሆኑ ተጠቃሚዎች እና ከፍተኛ መረጋጋት ለሚፈልጉ ሰዎች ትልቅ ጥቅም አለው።

የእግር ጣት መሬት አለመንካት (Toe Clearance) ሰው ሠራሽ እግርን በመጠቀም ሲራመዱ ከሚያጋጥሙ ትልልቅ ፈተናዎች አንዱ እግርን ወደ ፊት ሲያራምዱ የእግር ጣት መሬት እየተጋጨ መውደቅን ማስከተሉ ነው። የ4-ባር ጉልበት ግን ሲታጠፍ የሰው ሠራሽ እግሩን አጠቃላይ ርዝመት በትንሹም ቢሆን እንዲያጥር ያደርገዋል። ይህ አሠራር እግር ወደ ፊት ሲራመድ ከመሬት ጋር እንዳይጋጭ በቂ ከፍተት ይፈጥራል፣ ይህም የመደናቀፍ አደጋን በከፍተኛ ሁኔታ ይቀንሳል።

ለመቀመጥ ምቹ እና የተፈጥሮ ገጽታ ያለው መሆን

sample Data from Magazine source



“አለመኖር የሰው ልጅ ቀዳማዊ ፍርሃት ነው። መኖርም የሚጀምረው አለመኖርን ከመስጋት ነው።” ብሎ ወደ ተማሪዎቹ ፊት ተመለከተ። የመማሪያ ክፍሉ እንደ ሲኒማ ቤት ባለ ቅርጽ የተሰራ በመሆኑ ተማሪዎቹ አስተማሪውን አቆልቁለው ለማየት ይመቻቸዋል። መምህሩም የመምህርነት ትወናውን ከታች ወደ ላይ ተማሪዎቹን እያየ ይከውናል። ተማሪዎቹ አስተማሪ አቆልቁሎ ማየቱን ከመማሪያ ክፍሉ አቀማመጥ በበለጠ የወደዱት ይመስላል። አቆልቁሎ ማየት ተመልካቹን ታላቅ፣ የሚታየውን ደግሞ ታናሽ ስለሚያስመስል። በአሁን ዘመን ከማስመሰል በዘለለ፤ ነገርን የመረዳት ወይም የማወቅ ፍላጎት ደግሞ ጠፍቷል።

መምህሩ ዶክተር ደስታ ይባላል። ቀጥ ብሎ የተሰደረ አፍንጫና ጎላ ጎላ ያሉ ዐይኖች ለቀይ ዳማ ፊቱ ግርማን አላብሰውታል። ተክለ ሰውነቱ ረዘም ያለ ሆኖ ደልዳላ ነው። የእጅና የእግር ጡንቻዎቹ በትክክል መስመራቸውን ይዘው የፈረጠሙ ናቸው። ከደረቱ ወጣ ያለ ሆኖ፤ ሆዱ ከጀርባው ሊጣበቅ ትንሽ ነው የሚቀረው።

Sample Data from print out source

## መንፈስ ቅዱስ

ከሦስቱ አካላት (ከአብ፣ ከወልድ፣ ከመንፈስ ቅዱስ) አንዱ መንፈስ ቅዱስ ነው። መንፈስ ቅዱስ ከአብ የሠረፀ (የወጣ) ነው (ዮሐ 15÷26፣ ዮሐ 14÷26) ። ይህ ሲባል ግን ከአብና ከወልድ "ያንሳል።" ወይም "ይበልጣል።" ማለት አይደለም- ሦስቱም አኩል ናቸው አንጂ። የሦስቱም አካላት ስም በአንድነት መጠራቱም ይህንኑ ለማጠየቅ ማስረጃ ይሆናል (ማቴ 28÷19) ። ቅዱስ ጳውሎስም "የጌታ የኢየሱስ ክርስቶስ ጸጋ፣ የእግዚአብሔር ፍቅር፣ የመንፈስ ቅዱስ ኅብረት ከእናንተ ጋር ይሁን!" ብሎ ለቆሮንቶስ ክርስቲያኖች በመልክቱ መጨረሻ ተናግሯል (2ቆሮ 13÷14) ። መንፈስ ቅዱስ የራሱ የሆነ ፍጹም ገጽ፣ ፍጹምም አካል አለው። የአብና የወልድ የባሕርይ ሕይወታቸው ነው (ሮሜ 8÷9፣ ገላ 4÷6) ። አምላክ ነው። ከመከራ ሥጋ ከመከራ ነፍስ የሚያድን አዳኝ ጌታም ነው። የቤተ ክርስቲያን ምሥጢራት ሁሉ የሚከናወኑትም በእርሱ በመንፈስ ቅዱስ ማሕተምነት ነው።

**የመንፈስ ቅዱስ ሥራ**- መንፈስ ቅዱስ ፈጣሪ ነው። ጻድቁ ኢዮብ "የእግዚአብሔር መንፈስ ፈጠረኝ።" (33÷4) እንዳለ፣ ቅዱስ ዳዊትም "መንፈስህን ትልክሰህ ይፈጠራሉም።" (መዝ 103÷3) ብሎ እንደመሰከረ በእውነት መንፈስ ቅዱስ ፈጣሪ ነው (ዘፍ 1÷2፣ ኢዮ 26÷13፣ ዕብ 9÷14) ። አብ ወልድና መንፈስ ቅዱስ በፍቃድ አንድ በመሆናቸው የእግዚአብሔርነት ሥራ ሁሉ የመንፈስ ቅዱስም ሥራ መሆኑን እንረዳለን። ስለዚህም ነው መጽሐፍ ቅዱስ መንፈስ ቅዱስ መጋቢ፣ ለአገልግሎት ጠሪ፣ መልካሙን መስካሪ እንደሆነ የሚመስክረው(ሐዋ 8÷2፣ ሐዋ 13÷2፣ ሐዋ 2÷11፣ 1ጢሞ 4÷1፣ ራዕ 2÷7) ። በሌላም በኩል ቅዱስ ጳውሎስ "ለጌታ ቀን የታተማችሁበትን ቅዱሱን መንፈስ አታሳዝኑት።" በማለት መንፈስ ቅዱስ ኃጢአት በመሥራት ከፈቃዱ ስንወጣ እጅግ እንደሚያዝንብን አስተምሯል። ከዚህም በተጨማሪ መንፈስ ቅዱስ ሕይወትን ሰጪ፣ የመጽናናት ቃልን የሚለግስ አጽናኝ እንደሆነ ቤተክርስቲያን ታስተምራለች (ዘፍ 1÷2፣ ህዝ 36÷7፣ ዮሐ 16÷8፣ ማር 13÷11፣ 1ቆሮ 12÷1፣ ኢሳ61÷1፣ ሉቃ 24÷44፣ ሮሜ 15÷13) ። የርስቱ ወራሾች የምንሆነውም የእግዚአብሔር ልጆች እንድንሆን በጸጋው



Sample Data from book




Sample Data from other source